Calhoun
Institutional Archive of the Naval Postgraduate School

Calhoun: The NPS Institutional Archive

Theses and Dissertations | Thesis Collection

2005-12

The effect of high speed vessel operations on ship's crew and embarked personnel aboard HSV-2 SWIFT in the areas of motion sickness and motion induced task interruptions

Diaz, Alvaro

Monterey, California. Naval Postgraduate School

DUDLEY KNOX LIBRARY

Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

THE EFFECT OF HIGH SPEED VESSEL OPERATIONS ON SHIP'S CREW AND EMBARKED LANDING FORCE PERSONNEL ABOARD HSV-2 SWIFT IN THE AREAS OF MOTION SICKNESS AND MOTION INDUCED TASK INTERRUPTIONS

by

Gerald P. Lorio

Alvaro Diaz

December 2005

Thesis Advisor:	Michael McCauley
Co-Advisor:	Lyn Whitaker
Second Reader:	Arnold Buss

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 2005	Master's Thesis

4. TITLE AND SUBTITLE: The Effect of High Speed Vessel Operations on Ship's Crew and Embarked Personnel Aboard HSV-2 SWIFT in the Areas of Motion Sickness and Motion Induced Task Interruptions	5. FUNDING NUMBERS
6. AUTHOR(S) Gerald P. Lorio Alvaro Diaz	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
Approved for public release; distribution is unlimited	

13. ABSTRACT (maximum 200 words)

The Navy's use of high speed vessels such as HSV-2 SWIFT has raised questions of the effects of high speed motion on the ability of personnel to perform assigned duties. Performance degradation may occur during periods of excessive ship motion because of extreme motion sickness or periods of frequent task interruptions. With the use of high speed vessels expected to increase in the near future with the Littoral Combat Ship program, the issue of high speed motion effects on personnel becomes operationally relevant.

This study will take a two part approach to analyze the effects of high speed motion: the motion sickness of SWIFT's crew and military passengers, and interruptions of task performance caused by vessel motion to critical watch stations. For the first part, statistical analysis will be used to determine relationships between ship motion and motion sickness. For the second part, modeling and simulation will be used to determine if there are watch stations that may be affected by varying levels of motion induced task interruptions. From this analysis, guidelines may be produced to describe the expected levels of motion sickness in personnel as well as watch stations in which personnel may have difficulties performing assigned duties.

14. SUBJECT TERMS	15. NUMBER OF PAGES
HSV, JHSV, Motion Sickness, MII, MITI, Motion Induced Task Interruptions, LCS, High Speed Vessel Motion, MSAQ	225
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

THE EFFECT OF HIGH SPEED VESSEL OPERATIONS ON SHIP'S CREW AND EMBARKED PERSONNEL ABOARD HSV-2 SWIFT IN THE AREAS OF MOTION SICKNESS AND MOTION INDUCED TASK INTERRUPTIONS

Gerald P. Lorio
Lieutenant, United States Navy
B.S., United States Naval Academy, 2000

Alvaro Diaz
Lieutenant, Chilean Navy
M.S., Naval Engineering School, 2000

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
December 2005

Authors: Gerald P. Lorio
Alvaro Diaz

Approved by: Michael McCauley
Thesis Advisor

Lyn Whitaker
Co-Thesis Advisor

Arnold Buss
Second Reader

James Eagle
Chairman, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Navy's use of high speed vessels has raised questions of the effects of high speed motion on the ability of personnel to perform assigned duties. Performance degradation may occur during periods of excessive ship motion because of extreme motion sickness or periods of frequent task interruptions. With the use of high speed vessels expected to increase in the near future with the Littoral Combat Ship program, the issue of high speed motion effects on personnel becomes operationally relevant.

This study will take a two part approach to analyze the effects of high speed motion: the motion sickness of HSV-2 SWIFT's crew and military passengers, and interruptions of task performance caused by vessel motion to critical watch stations. For the first part, statistical analysis will be used to determine relationships between ship motion and motion sickness. For the second part, modeling and simulation will be used to determine if there are watch stations that may be affected by varying levels of motion induced task interruptions. From this analysis, guidelines may be produced to describe the expected levels of motion sickness in personnel as well as watch stations in which personnel may have difficulties performing assigned duties.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

To the great player that couldn’t make it to this world…

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The Navy's use of high speed vessels has raised questions of the effects of high speed motion on the ability of personnel to perform assigned duties. Performance degradation may occur during periods of excessive ship motion because of extreme motion sickness or periods of frequent task interruptions. With the use of high speed vessels expected to increase in the near future with the Littoral Combat Ship program, the issue of high speed motion effects on personnel becomes operationally relevant.

This study will take a two part approach to analyze the effects of high speed motion: the motion sickness of HSV-2 SWIFT's crew and military passengers, and interruptions of task performance caused by vessel motion to critical watch stations. For the first part, statistical analysis will be used to determine relationships between ship motion and motion sickness. For the second part, modeling and simulation will be used to determine if there are watch stations that may be affected by varying levels of motion induced task interruptions. From this analysis, guidelines may be produced to describe the expected levels of motion sickness in personnel as well as watch stations in which personnel may have difficulties performing assigned duties.

The study of motion sickness in ship's crew and other embarked personnel yields a relationship between motion sickness, wave heights, and exposure times: in general, the larger the waves, the more severe the motion sickness. With longer exposure times, personnel will adapt to ship motion, which decreases motion sickness. An Operations Manual is produced for use by commanders to predict levels of motion sickness given expected wave heights and exposure times.

The study of motion induced task interruptions yields a relationship between performance degradation and the lengths of interruptions, as well as their frequencies. For most watch stations, the longer interruptions are and the

more frequent they become, the more a watch stander is degraded in their task performance. Given the model used to simulate these conditions, the degradations to task performance are not significant enough to cause watch standers to fail to perform their duties.

I. INTRODUCTION

The Navy's use of high speed vessels such as HSV-2 SWIFT (Figure 1) has raised questions of the effects of high speed motion on the ability of personnel to perform assigned duties. Performance degradation may occur during periods of increased ship motion because of severe motion sickness or periods of frequent task interruptions. With the use of high speed vessels expected to increase in the near future with the Littoral Combat Ship program, the issue of high speed motion effects on personnel becomes operationally relevant.

Figure 1. HSV-2 SWIFT (USMC High Speed Connector Website).

A. BACKGROUND

1. High Speed Vessels and the Littoral Combat Ship Concept

High speed vessels have been used in civilian applications for many years as commercial vehicle and passenger transportation assets. These vessels bring a unique set of operational characteristics that make them invaluable to commercial industry (Gourley and Scott, 2005). Hull designs such as catamarans or trimarans allow high speed vessels to transit efficiently at high speeds to optimize transit time while maintaining or reducing fuel costs. These designs also give the vessels a relatively shallow draft which allows them access to a larger of number of ports. High speed vessels have excellent maneuverability which increases safety and reduces the need for harbor tug boats. This maneuverability, combined with the ability to rapidly start and stop main engines, allows for rapid entrance and exits of ports. Many high speed vessels also have self contained on and off load ramps that reduce the need for pier services, thus lowering operating costs and allowing for access to under-developed ports.

The United States (U.S.) military has taken notice of high speed vessels (HSVs) and their expanding use in civilian applications. The military is also interested in exploring the advantages that these vessels could bring, with few modifications, to operations conducted in littoral areas (Memorandum for Joint Speed Vessel (JHSV) Analysis of Alternatives (AoA) Study 2005). These advantages include rapid intra-theater medium payload surface lift for battalion sized units, rapid deployment of forces and supplies to minor and degraded ports, a high degree of maneuverability in close operating situations, self-contained onload and offload capabilities, and reduced development, production and operating costs. These factors can assist operations such as those in support of the Global War on Terrorism and in theater security by providing capabilities that span theaters of operations. Additionally, they will aid in bringing the seabasing concept, which involves projecting military power to a land

operating area from the sea, closer to a reality by providing high speed connectors to rapidly transport troops and material from forward at-sea operating bases to the shore.

In 2001, the U.S. signed a lease for three high speed vessels from the Australian ferry company, Incat Group (Gourley and Scott, 2005). The first of these vessels was the HSV-X1 JOINT VENTURE, placed in service in October, 2001, to serve as a proof of concept and evaluation platform for trials and demonstrations. Next was the TSV-1X SPEARHEAD, placed in service in November of 2002 and operated by the U.S. Army as an advanced concept technology demonstrator. HSV-2 SWIFT, placed in service in August of 2003, was leased to serve as an interim replacement of the mine countermeasure ship USS INCHON. SWIFT is also being utilized to explore concepts and capabilities of high speed designs for the Littoral Combat Ship (LCS) program. These three ships have taken part in many operations during the past few years to include Enduring Freedom, Iraqi Freedom, Joint Task Force Horn of Africa operations, Operation Cobra Gold, Operation Talisman Sabre, the tsunami relief effort in Indonesia, and the Hurricane Katrina relief effort in New Orleans. They have served as command and control platforms, special operations bases, helicopter coordination and refueling platforms, high speed inter-theater troop transports, and long range cargo transports. In addition to the HSVs leased from the manufacturer, Incat, the Austal Ships' *Westpac Express* was assigned in February 2002 to the U.S. Military Sealift Command on a three year lease in support of the Third Marine Expeditionary Force (MEF) in Okinawa, Japan. The *Westpac Express* has served as transport for the Marines and their supplies in the Pacific area of operations for over three years.

Both the U.S. Navy and the Army realized the advantages that the leased HSVs provide to current forces and began to develop separate service requirements for obtaining vessels built in, and owned by the United States (Memorandum for Joint Speed Vessel (JHSV) Analysis of Alternatives (AoA) Study). In 2004, the effort to assess the need of HSVs for the U.S. military resulted in a memorandum of intent that transferred the responsibility of

developing an acquisition program for Joint High Speed Vessels (JHSVs) to the U.S. Navy. In May 2005 the navy set in motion an acquisition program to result in a contract by fiscal year 2008.

In addition to the benefits that JHSVs offer to seabasing and high speed lift, they are also being utilized in the development of the LCS program to explore the capabilities of catamaran type hull forms as well as the implications of high speed operations on embarked personnel and equipment. The LCS will be a relatively small and high speed combatant that is able to operate in shallow waters, less than 20 feet deep, and at up to speeds of 50 knots (PEO Ships Official U.S. Navy Website). LCS also will be much less expensive than current destroyer (DD(x)) and cruiser (CG(x)) designs. The missions of the LCS will include: low speed sustained support to forces ashore, high speed operations against littoral threats such as small boats and submarines, and high speed insertion and extractions of landing forces. These missions bring the need for a fast and efficient ship design that has the capability to reach littoral regions around the world quickly. Two preliminary designs have been contracted for the development of the first four production units (Naval Technologies LCS Website). One is a high speed semi-planing monohull design (Figure 2) that that has been contracted to Lockheed Martin for two ships. The keel for the LCS-1, to be christened the USS FREEDOM, was laid in June 2005 to be completed in late 2006 and commissioned in 2007. Construction for the LCS-3, Lockheed Martin's second ship is scheduled to begin in 2006 for a 2008 commissioning date. The second contract for two ships was awarded to General Dynamics. The General Dynamics ship will have a high speed trimaran hull design that incorporates a slender, stabilized monohull (Figure 3). Construction of the General Dynamics designs, LCS-2 and LCS-4, is scheduled to begin in 2006.

Figure 2. Lockheed Martin Monohull LCS Design (PEO Ships Official U.S. Navy Website).

Figure 3. General Dynamics Trimaran LCS Design (PEO Ships Official U.S. Navy Website).

There is little doubt that the current JHSVs and the future LCS will bring many previously unobserved advantages to littoral operations, but there have also been reports from embarked personnel in the JHSVs of extreme cases of motion sickness and decrements to task performance due to motion induced interruptions. These factors may not be critical when transferring civilian passengers from one place to another as in the previous role of these high speed

ferries, but when converting these designs to military applications, they have the potential to become important. For example, if a JHSV is tasked to transport a landing force to a potentially hostile area, a likely mission according to the seabasing concept, extreme motion sickness may affect the combat readiness of the troops that make up that force. Also, if the crew of a future LCS is operating in a potentially hostile littoral area, degradation of crew performance may hinder their capability to effectively fight the ship. Situations such as these cause concern as to the operability of high speed surface ships in combat environments, especially since the Navy is currently taking the steps to procure more JHSVs and is in the process of developing and ordering several LCS class ships.

2. Motion Sickness

Cases of motion sickness can be traced back thousands of years to the ancient Greeks. Writers of mythology wrote about sea sickness experienced by people traveling by boats. More over, “naus” is the root of the word “nausea,” which in Greek means “ship,” thus associating this effect to people that traveled by sea and with a clear allusion to the vomiting effect that this form of transportation caused (Griffin, 1990).

Motion sickness is the normal response of the human body to some stimulus of motion, not a medical problem which is what many tend to associate with the word sickness. One of the most common effects of motion sickness is vomiting which may lead to medical treatment due to the loss of a large volume of fluids; but this treatment is a consequence of a symptom of motion sickness, not from the motion sickness itself. Motion sickness is not fatal and is normal to human beings, but as Sanders and McCormick state: “Although motion sickness may never kill us, there are times when we wish it would” (Mansfield, 2005).

The most common and well known effect of motion sickness is vomiting, but there is a large list of symptoms that are normally experienced before

vomiting. Pallor, cold sweating and nausea are the most common symptoms reported by most studies (Money, 1970; Griffin, 1990). These symptoms affect every individual in a different way, depending of how susceptible a person is to motion, and not all of the symptoms are experienced by a every individual. But given that they arise, they normally follow the sequence of stomach discomfort followed by nausea, pallor, sweating, an increase in salivation, eructation, and finally vomiting. Although vomiting may bring relief, if the exposure to motion continues, the discomfort will rise again until some kind of adaptation is developed by the individual (Benson, 1984).

Studies have determined that there are two factors that cause motion sickness: the physiological response to motion, and the psychological response to the situation in which the motion is occurring (Mansfield, 2005).

The psychological part has to do with the will of a person to overcome the motion sickness symptoms and to be able to perform normally. Studies have shown that the symptoms of motion sickness decrease when subjects shift their attention to something else. However, when they cease to focus, the symptoms increase again (Mansfield, 2005).

To understand the physiology of motion sickness, a brief look into the three systems that give the body the inputs of motion to which it is exposed is necessary. Figure 4 shows a simple diagram of the human Vestibular System, the first sensory system. It has two subsystems located in the inner ear that play a significant role in motion sickness: the otoliths and the semicircular canals, which are both located inside the Middle Ear. The otoliths detect linear acceleration and the semicircular canals detect angular acceleration. These sensors transform the motion that a body is exposed to into information that is processed by the brain. Although these are not very accurate or sensitive and are not capable of sensing frequency and movement as good as a modern accelerometer, these are why a person gets disoriented and eventually motion sick (Griffin, 1990).

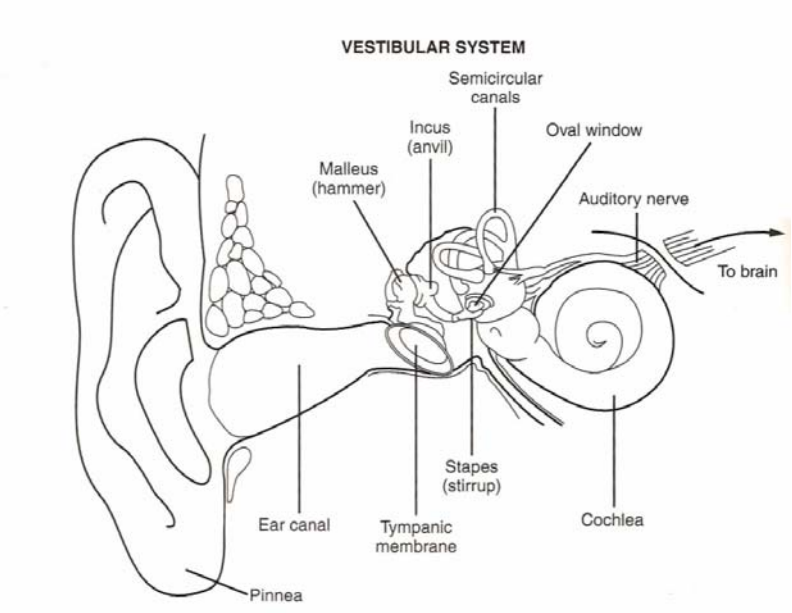

Figure 4. Human Vestibular System (Wickens, Lee, Liu, and Becker 2004)

The second sensory system involved in motion sickness is the visual system. The eyes send the brain an image of the physical surroundings, which is processed into a spatial orientation by the brain. The third sensory system involved in giving motion inputs to the brain is the proprioceptive or somatic system. It contains sensors that record changing pressures around the body which include the skin, muscle and joints (Stevens and Parsons, 2002). These three sensory systems combine to provide the brain with inputs to orient the body and maintain balance in the presence of motion.

The causes of motion sickness are not exactly known, but *sensory conflict*, is the most accepted theory (Reason and Brands, 1975). This theory states that when a conflict occurs between what a person sees and what they feel, over time the result is motion sickness.

When a person is exposed to motion, their brain receives inputs from the three sensory systems. If we add a fourth input, “control,” which can be understood as the motion that the brain expects the body to be exposed to as result of an action, there are four inputs to the brain that can affect the human response to motion. When these four inputs cause a perceived conflict in the

human brain, the result is the development of motion sickness. One typical example of this is when passengers in a car get motion sick while they are reading. Drivers almost never gets motion sick, because they are in control so they can anticipate when the car will turn right or left, and when their bodies sense the movements, they are in accordance with their perceived actions. On the other hand, the passengers who are reading have a conflict between what they see and what they sense because their eyes are focused on the book which is relatively still, and what their body senses is motion which conflicts with what the visual stimuli implies what they should be feeling. The result of this conflict is motion sickness.

Two types of conflict, or sensory mismatch, can be experienced: intersensory or intrasensory (Griffin, 1990). Intersensory conflict is produced when two different sensors receive inputs that are not correlated. Within the intersensory conflicts, two types can be distinguished:

- Type 1 Two systems sense motion, but of incompatible kind and not in accordance with the expected response.
- Type 2 One system senses motion but the other doesn't.

Intrasensory conflict is produced within the vestibular system when the otoliths and semicircular canals don't agree on their inputs. Intrasensory conflicts can also be of two types:

- Type 1 The otoliths and the semicircular canals sense motion but of an incompatible kind.
- Type 2 The otoliths sense motion but the semicircular canals don't, or vice versa.

Exposure to motion can cause conflicts between our sensory systems which in most cases produce motion sickness. However, the brain correlates the sensory inputs based on expected reactions to these inputs, so with time of exposure, the brain is adapting the expected reaction to the conditions to witch the body is exposed. This adaptation decreases the conflict. Many studies in motion sickness have addressed this issue and all agree that adaptation is

expected for most individuals, excluding around 5% who will never experience adaptation regardless of exposure time (Money, 1970; Benson, 1999; McCauley et al, 1976; Crossland, 1998). Figure 5 displays the adaptation curve obtained by McCauley et al (1976) for people exposed to five daily two-hour periods of motion in a simulator. The criterion, motion sickness incidence (MSI) was defined as the proportion vomiting among the sample due to motion sickness. Figure 6, develop by Crossland (1998) displays a similar adaptation curve for people who traveled by ship. Both curves show the adaptation phenomenon independent of how the motion sickness was induced, ship or simulator. The curves show that the brain adapts the expected reactions to the movement stimuli, thus decreasing the MSI among people.

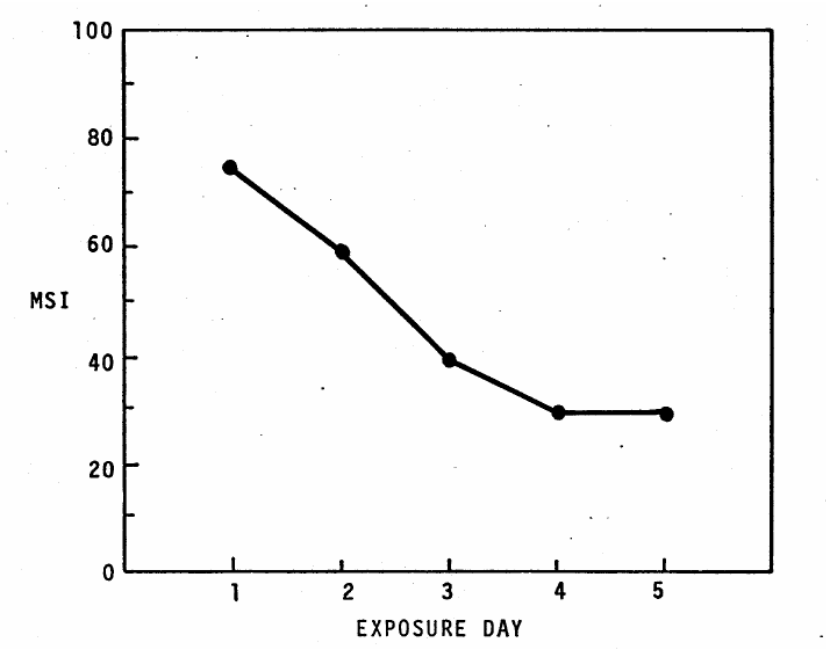

Figure 5. MSI Adaptation curve (McCauley et al 1976)

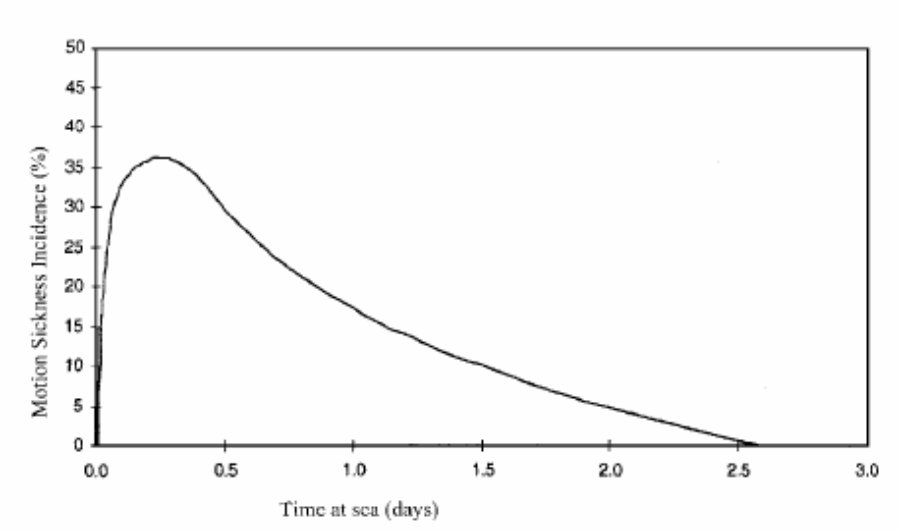

Figure 6. MSI Adaptation curve (Crossland 1998)

Predicting the incidence of motion sickness is difficult given that the factors involved differ from person to person and even within individuals, vary depending on situations and experience levels (Smart, 2002). Among the models already developed for predicting motion sickness, two are often used as a basis for studies: first, the McCauley and O'Hanlon in (1974) and McCauley et al (1976) Motion Sickness Incident (MSI), and second, the Lawther and Griffin (1987) and (1988), Vomiting Incident (VI).

The McCauley and O'Hanlon (1974) model, extended in McCauley et al (1976), was developed in a sea motion simulator by exposing people to a single frequency sinusoidal vertical motion. The McCauley model states that the principal component of the MSI was the vertical component of acceleration that subjects were exposed to with almost no MSI effects due to pitch and roll. They found that the maximum severity of MSI was produced at a frequency of 0.167 Hz. Figure 7 displays a three dimensional graph with the percentage of MSI versus frequency and RMS acceleration (g).

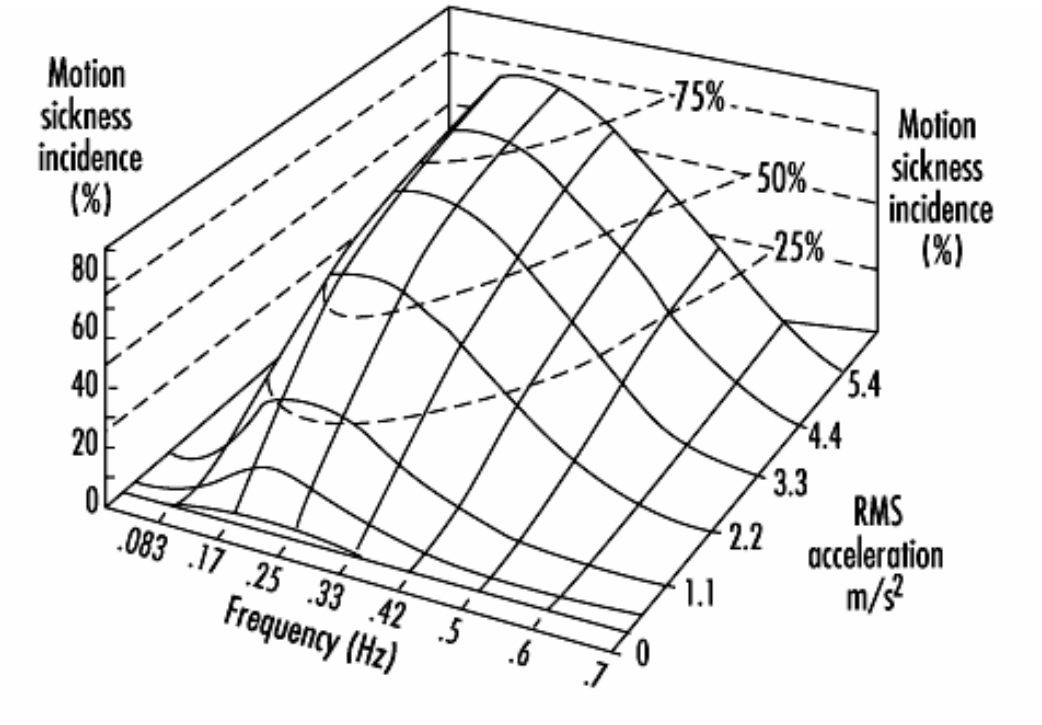

Figure 7. MSI incident of McCauley at al (1976)

The Lawther and Griffin (1987) model uses the same parameters as the McCauley et al (1976) model to predict motion sickness vomiting, but they identify it by Vomiting Incident (VI) to emphasize the difference in the methodology of obtaining the result, given that VI experiments were conducted onboard a car ferry. The results for the VI model were similar to those of the MSI model: the biggest factor that affected motion sickness was vertical acceleration. Figure 8 shows the results from Lawther and Griffin using experimental data from McCauley et al. Lawther and Griffin (1987) also found that, while pitch and roll didn't play a significant roll in the model, they shouldn't be entirely discounted (Stevens and Parsons, 2002).

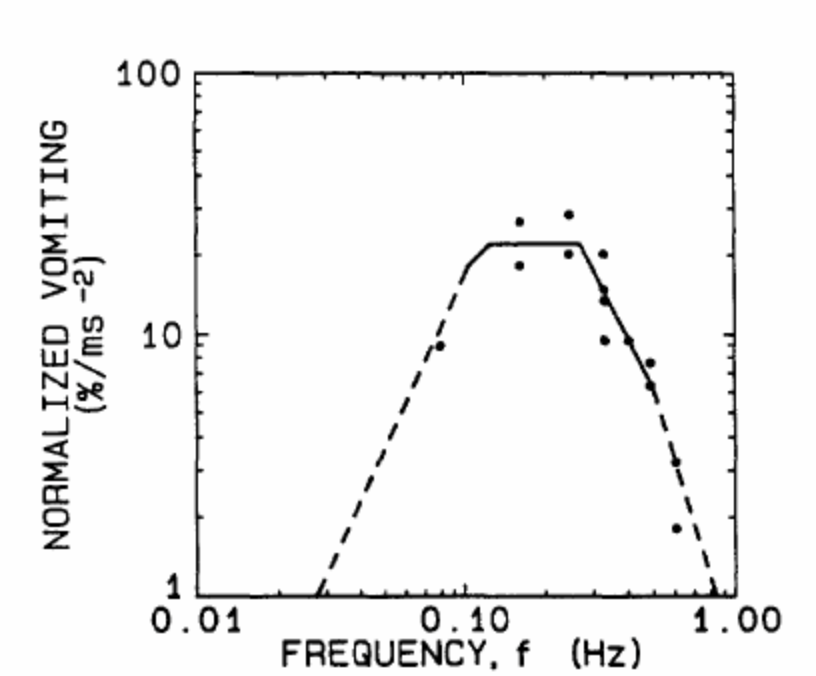

Figure 8. Lawther and Griffin (1987) model plot with McCauley et al (1976) data.

The studies performed by both O'Hanlon & McCauley in 1974 and Lawther & Griffin in 1987 form the basis for the contemporary standards to predict motion sickness. The first standard is part of the US military standards (MIL-STD-1472) and the second standard is currently used in the International Standard Organization (ISO) and in the British Standard Organization (BS) to predict incidence of motion sickness (ISO2631, BS2631). In these publications, a "Motion Sickness Dose Value" (MSDV) is defined in order to predict the percentage of people who will likely vomit after being expose to vertical accelerations of a certain magnitude for certain times.

The two motion sickness models previously discussed are based on the percentage of people who vomit after being exposed to motion of varying frequencies and amplitudes. However, as stated before, motion sickness has several symptoms that occur prior to vomiting. There have been many studies to determine an effective way to measure the extent of motion sickness in an individual. The results of these have yielded several motion sickness scoring systems: the Pensacola Diagnostic Index (PDI) (Graybiel, Wood, Miller, and

Cramer, 1968) the Motion Sickness Questionnaire (MSQ) (Kellogg, Kennedy, and Graybiel, 1965), and the Motion Sickness Assessment Questionnaire (MSAQ) (Gianaros, Mordkoff, Muth, Levine, and Stern, 2001). The motion sickness scoring system that was selected for this study is the latter, MSAQ. The MSAQ was chosen in favor of the others because it utilizes a four dimensional approach to categorize the effects of motion on the human body. The four dimensions that the MSAQ categorizes and gives a subscale score are Gastrointestinal, Central, Peripheral and Sopite-related. In this study only the combined score for each individual is used, but the MSAQ scores provide flexibility for any following studies.

3. Motion Induced Task Interruptions

There are two problems due to vessel motion effects in a naval environment: motion sickness and other biodynamic problems (Colwell, 1989). Other biodynamic problems can be broken into three smaller areas: motion induced fatigue (MIF), which is the tiring effect that motion has on individuals over prolonged exposure due to lack of sleep and other stresses due to vessel motion; whole body vibration (WBV), which consists of the physiological effects of different frequencies on the human body in areas such as vision and motor control, and motion induced interruptions (MII), which occur when local motions cause an individual to lose balance and cause the task being performed to be interrupted. This portion of the study will focus on a generalized form of motion induced interruption called motion induced task interruptions (MITIs).

The term “Motion Induced Interruption” (MII) was first introduced by Baitis (1984) while investigating the effects of ship motion on a flight deck crew onboard a FFG-7 Oliver Hazard Perry class frigate during helicopter operations. The MIIs analyzed in that study were broken into three categories: tipping, sliding, and lifting off the deck. Considerable research and analysis has been put forth in the study of MIIs because of their effects on human task performance and

the implications that biodynamic vessel designs can have on preventing them. Studies of the three categories of MIIs have resulted in mathematical models for the effects of motion on a rigid human model. The result of this research was a "stick figure" model that describes the effects of ship motion on a rigid human frame standing somewhere on a ship. The "stick figure" had a certain center of gravity, stance width, and was not capable of adjusting its posture to compensate for ship motion.

Analysis of this model yielded four equations that were formalized by Graham (1990) to describe how often the "stick figure" would tip over to port, starboard, fore, or aft in varying pitch and roll conditions. Graham, Baitis and Meyers (1991) later developed the MII concept to include a sliding motion which expanded the previous model to allow the "stick figure" to slide to port or starboard as well as to tip over. These models were further refined in 1992 (Baitis, Holcombe, Conwell, Crossland, Colwell, Pattison, and Strong, 1995) and exposed to experimental validation in the ship motion simulator at the Naval Biodynamic Lab. The validation showed that the equations used to predict lateral tipping tended to over-predict the occurrence of MIIs due to the experimental subjects' ability to correct for motion by lowering their stances and, thus stabilizing their frames. Further analysis revision yielded experimental values for coefficients in the MII equations that allow them to give a reasonably accurate prediction of MII occurrence in tasks involving tipping (Baitis, Holcombe, Conwell, Crossland, Colwell, Pattison, and Strong, 1995).

The Navy's trend toward reduced manning highlights the importance of determining how often MIIs occur and how they degrade task performance. Fewer individuals will be available on ships to perform tasks so their performance of these tasks becomes critical (Stevens and Parson, 2002). To analyze performance, tasks are broken down into four categories: perceptual tasks which require audio or visual signal detection; motor tasks which require motor skills for manual tracking or to push buttons; cognitive tasks which require memory, reason, and more thought processes; and complex tasks which involve some combination of perceptual, cognitive, and motor tasks. Most tasks aboard a ship

are complex tasks such as making or responding to a radio call, tracking a contact on a RADAR system, plotting the ship's position, or changing the ship's course or speed. Similar complex naval tasks have been simulated in experiments at the TNO Human Factors Research Institute to study the effects of vessel motion on these tasks (Stevens and Parson, 2002). The results indicated that motion caused a reduction in information transfer in the tasks, but that the results of simulating complex tasks could not be transferred to other tasks. The complex tasks had to be broken down into their simpler components in order to apply analysis to comparable tasks.

The study of motor tasks is what led Baitis, Graham, and Meyers (1991) to the development of the MII equations, but limits their applicability to strict motor tasks. The MII equations also require detailed acceleration data and strict subject position information to correctly approximate the number of MIIs that will be experienced. The MIIs are directly derived from the tipping or sliding events that the equations predict. MITIs offer a more general approach to studying task interruptions caused by vessel motion. The MITI approach is to gather data about complex tasks aboard a vessel by observing personnel performing these tasks. The data includes the steps of the process, the stimuli that cause it, and the average times it takes different personnel to complete the tasks. Data about incidents of task interruptions are also gathered from observations aboard the same vessel. The interruption data includes the length of the interruption, the subject's position prior to the interruption, whether it be sitting, standing, walking, kneeling, or squatting, the sea conditions, and the speed of the vessel through the water. Each of these interruption events will also be correlated to the degree of peak vertical acceleration (g's) that led to the interruption. Vertical acceleration is particularly important for high speed vessels because of the slamming. For a specific ship type (hull design), this approach will yield an average interruption length, as well as the frequency of interruptions, given sea conditions and ship speed. Interruption length and frequency are two parameters that can then be used in an aggregate level task simulation model to determine degradations to complex task performance under varying conditions.

B. OBJECTIVES

This thesis will examine the effects of high speed vessel operations on the performance of embarked landing force personnel and the ship's crew. Two main areas of focus will be used to explore these two groups of personnel. The first is the study of the effects of high speed motion on an embarked landing force. This analysis will focus on determining if there is the potential for the degradation in the combat capability of the landing troops due to motion sickness after being embarked on an HSV and being exposed to ship motion. The second focus of study is on the potential for degradation of task performance in the ship's crew due to the effects of high speed operations in the form of motion sickness or motion induced task interruptions.

Additionally, this study will attempt to develop mission specific operational guidelines for the employment of JHSVs.

C. COURSE OF STUDY

The initial phase of this study was the data collection performed aboard HSV-2 SWIFT in two different stages. The first stage was conducted as the ship was returning from Operation African Lion with an embarked landing force of U.S. Marines. The research team issued motion sickness surveys to the Marines to be filled out every hour as to their current state of readiness and the estimated states of readiness of the Marines around them.

The second stage was conducted in a transatlantic crossing during which motion sickness surveys were issued to the crew to be annotated several times on a daily basis as to their state of readiness. Several hierarchical task analyses also were performed on tasks important to daily ship operations. Logs were kept on SWIFT's Bridge during both phases including ship's speed and heading, wave height and direction, and ride control usage. The ship also was fitted with sensors to measure accelerations on the three major axes. In addition to these measures, a portable usability lab was installed to study and record crew members' performance at completing hand-eye coordination exercises in

different sea states. The tasks were preformed on a computer and involved exercises that required the user to manipulate mouse or trackball input devices to perform scrolling, clicking, and text editing tasks.

The next phase of this study was to model the effects of high speed motion on motion sickness and motion induced task interruptions based on data from the SWIFT. A motion sickness score was derived to describe an embarked landing troop depending on ship motion conditions. This model may give commanders the capability, during the planning stage of a high speed movement operation, to estimate the condition their forces will be in upon arrival. This estimate may be based on transit speed, sea state, ship's heading, exposure time, or stability control use, or a combination of any of these factors. Part of the data will be used to determine the appropriate model and then, the remainder of the data will be used to validate it. The result of this study will be a first approach to an operating guideline regarding the effects of high speed transit on an embarked landing force.

The course of action in the area of motion induced task interruptions has two parts. The first part is to use the data collected in the HSV-2 transatlantic crossing to determine the rate of occurrence and duration of MITIs aboard the SWIFT. The rates of occurrence of MITIs can be obtained from the z-axis (vertical) acceleration data of the accelerometers that were placed in various locations on the ship. This z-axis data can then be fitted against relative wave direction, significant wave height, and ship's speed to determine rates of occurrence of MITIs given those factors. The durations of the interruptions can be determined from an analysis of video taken in several spaces in the ship. The occurrence of interruptions was noted in the videos and their durations were recorded along with the time, date, camera placement, subject's original position, and whether or not the subject was braced prior to the interruption. This interruption data can be fit to a multiple linear regression model along with ship's speed, relative wave direction, and significant wave height to determine the most likely duration of MITIs, given underlying conditions.

The second part of the study of task interruptions is to take the results from analyzing the MITI frequency and duration data from SWIFT's transatlantic crossing and use them as inputs to a task simulation model to determine if and when motion induced interruptions significantly degraded task performance. The model was developed in Simkit (Buss, 2002) and includes four major watch stations that can be subjected to varying ships missions and MITIs. The missions that are modeled are those that HSVs are currently called upon to complete as well as those that a future LCS may be tasked to do. They include: high speed transit in an open ocean environment such as that required for inter theater lift; high speed transit in a littoral environment such as required for the transfer of troops and personnel from a sea base to the shore; high speed anti-surface warfare (ASUW) in a littoral environment such as may be required in an operation in a littoral region; and naval surface fire support (NSFS) which may be a part of supporting a landing force in a littoral conflict. The utilization of each modeled watch stander can be determined after multiple simulation runs in each of the four missions under varying degrees of MITIs. These utilizations can then be used to determine whether or not a watch stander was significantly degraded in their ability to complete their assigned tasks. While this model is not a human watch stander, it may provide guidance to system designers and mission planners as to what missions and watch stations may represent areas of concern when operating at high speeds.

D. LIMITATIONS

There are several limitations that this study faces. One of the largest is that there is not a known comparative study conducted on a traditional monohull ship. An ideal comparative study would be conducted with a monohull ship that is similar in size to the SWIFT and is conducting similar tasks while operating side by side in the same seas. That procedure would provide insight as to whether the catamaran hull design of the SWIFT yields an increase in personnel

motion sickness or an increased degradation of task performance due to MIIs compared to the monohull design. While a comparative study would be ideal, it is not available and this study must focus solely on the data collected aboard the SWIFT.

Another limitation of this study is the data collection source. The MITI data was taken from the SWIFT, a JHSV that at the time of the data collection effort had no offensive weapons systems and no combat watch standers. The future LCS applications of this study should include analysis of a combat watch stander in order to explore the effects of MITIs on some of the possible mission areas, anti surface warfare and naval surface fire support. To compensate for this limitation, the Tactical Action Officer (TAO), one of the most tactically significant and often most burdened watches aboard a surface combatant, was modeled. The parameters and roles of the TAO were taken from current CG and DDG class combatants and modeled in the Simkit modeling tool. This model was run with the interruption input data provided by the SWIFT data in order to simulate what may be a combat watch stander aboard a future LCS. While no LCS class is currently operational, the modeled TAO is intended to be an approximation to what the actual TAO watch stander.

Another limitation lies in the conditions of the transatlantic data collection process. The ship was not in condition to support the researchers constantly because it was burdened with operational tasking. Therefore participants were exposed to a limited number of conditions during the transit. There was no chance to test the crew and embarked troops under all possible conditions of speed, exposure time, sea state, and ship's heading in order to have a more representative sample of the motion sickness scores.

There is a limitation in the variables that are available to related to the survey scores from the Marines and crew taken from the data collected on SWIFT. Previous studies indicate that vertical acceleration is the most influential variables that influences motion sickness (McCauley and O'Hanlon, (1974); McCauley et al, (1976); Lawther and Griffin, (1987) and (1988)). The collected

data does not include vertical acceleration for the Marine data and only includes segments of vertical acceleration for the crew data. However, the conditions that cause accelerations such as wave heights and ships speed are available and will be used to correlate with the motion sickness survey scores.

A final limitation is that many of the input parameters to the task simulation model are estimates. The Atlantic transit did not provide an opportunity to collect the data needed for all missions and tasks. Therefore, estimates were used for some mission data and statistical techniques were applied to explore the effects of varying these estimated parameters.

THIS PAGE INTENTIONALLY LEFT BLANK

II. METHODOLOGY

The data collected onboard the HSV-2 SWIFT was analyzed using several statistical and simulation modeling techniques. Different types of data were collected from the SWIFT under different circumstances, so all pertinent data sets were reviewed to determine which sets contained information most relevant to this study. Once this was determined, statistical models were generated for the data sets relating ship and wave conditions to motion sickness scores. The Simkit simulation model was then developed from hierarchical task analyses of key watch stations and from the relationships between ship motion and wave conditions and MITI length and frequency. Once the simulation model was complete, statistical techniques were applied to the model's output to develop linear regression models relating ship and wave conditions and MITI rates to the ability of each watch stander to complete assigned duties in each of four mission areas.

A. DATA SETS

Data sets that were considered for use in this study were collected in four different periods. Three of these data sets were collected while SWIFT was under other operational tasking, which limited the level of dedication that the ship was able to provide to the data collection efforts. The fourth data set was collected during SWIFT's sea keeping trials.

1. Data Sets from Operational Periods

a. Crew Motion Sickness, December 2004

In December of 2004, the SWIFT took part in mine warfare exercises in the Gulf of Mexico. During this exercise, 21 MSAQ questionnaires where distributed to the crew and 17 were returned completed. Unfortunately, no records were available of the sea state, ship's speed, or wave direction. Given

that these variables can greatly affect the results of the MSAQ questions, but were not recorded, this data set was not useful for this study.

b. U.S. Marine Motion Sickness, April 2005

SWIFT transported nearly 200 U.S. Marines and their equipment from Morocco to Spain at the end of OPERATION AFRICAN LION. MSAQ questionnaires where distributed to the Marines in order to collect data that pertained to the motion sickness of embarked landing force personnel. The transit from Morocco to Spain was conducted over a 24 hour period in which the ship experienced high sea states and rough weather. Only 22 MSAQ surveys were completed to some extent by the Marines on board. Sea states and vessel operating parameters also were recorded.

c. Crew Motion Sickness, April 2005

After the SWIFT disembarked the Marines from OPERATION AFRICAN LION, it proceeded on a transatlantic crossing from Rota, Spain to Little Creek, Virginia. The authors as well as three personnel from the Naval Surface Warfare Center in Panama City, FL embarked for the transit which was conducted in nine days with two stops for fuel, one in the Canary Islands and one in Bermuda. MSAQ questioners were given to 25 members of the crew and logs were maintained in the bridge to record sea state and vessel operating data on an hourly basis. Seventeen of the MSAQ questionnaires were completed to some extent by the end of the transit and their scores were related to the conditions in which they where recorded. A limiting factor of this data collection period was that one of SWIFT's four main engines was inoperable. The ship also had a cargo of eight light amphibious vehicles as well as some additional equipment left behind by the Marines. The combined effects of cargo weight and an engine casualty limited the average speed of the transit to less than 24 knots; therefore, there was no data available during this period to explore the effects of speeds higher than 24 knots.

d. Crew Motion Induced Task Interruptions, April 2005

MITI data was also collected during SWIFT's transatlantic crossing following OPERATION AFRICAN LION. Vessel motion data was recorded by accelerometers installed onboard the SWIFT in several areas including the galley, bridge, and near the C4I spaces. In addition to the accelerometers, several video cameras were installed to document incidents of interruptions during daily operations. The resulting raw data from this collection is currently under analysis by the Naval Surface Warfare Center, (NSWC) Panama City, research team; however, several periods of z-axis acceleration data and correlating MITI analysis from video footage were made available to the NPS research team for this study (see APPENDIX A). In addition to this data, several task analyses were performed of critical shipboard tasks by the NPS researchers. The times to complete these tasks were also recorded to facilitate the construction of the task model in Simkit (see APPENDIX D).

2. Data from HSV-2 Sea Trials

Sea keeping trials were conducted on the HSV-2 SWIFT between February and May of 2004. These trials tested SWIFT under various conditions in order to determine its suitability for future naval operations. MSAQ questionnaires were distributed to the crew during the second stage of the trials in the North Atlantic. Of these, 19 were returned completed over a period of 14 days. While the MSAQ scores were available for study, the records of ship motion conditions were not available from NSWC. Without sea state conditions or ship operating conditions such as speed, the MSAQ results are of very little use in this study.

B. MOTION SICKNESS METHODOLOGY

1. Dependent Variable Motion Sickness Score

In order to quantify the motion sickness that a person is experiencing during a particular period of exposure to motion, the Motion Sickness Assessment Questionnaire (MSAQ) was used. Even though the MSAQ was chosen because it explores motion sickness in four dimensions, only the total score was used in this study, leaving the flexibility of the other dimensions for a later study.

The MSAQ questionnaire is composed of 16 questions that have scores ranging from one to nine (Table 1). Lower scores indicate few symptoms from motion sickness and higher scores indicate more severe symptoms. Each participant records their answers to the 16 questions at the same time.

Table 1. MSAQ Scores

Not at all								Severely
1	2	3	4	5	6	7	8	9

1 Ifell Sick to my stomach (G)
2 I fell faint-like (C)
3 I fell annoyed/irritated (S-R)
4 I fell sweaty (P)
5 I fell queasy (G)
6 I fell lightheaded (C)
7 I fell drowsy (S-R)
8 I fell clammy/cold sweat (P)
9 I fell disoriented (C)
10 I fell tired/fatigued (G)
11 I fell nauseated (G)
12 I fell hot/warm (P)
13 I fell dizzy (C)
14 I fell like I was spinning (C)
15 I fell as if I may vomit (G)
16 I fell uneasy (S-R)

$$CScore=\left(\frac{\sum_{C}\text{Questions}}{45}\right)*100$$

$$GScore=\left(\frac{\sum_{G}\text{Questions}}{36}\right)*100$$

$$PScore=\left(\frac{\sum_{P}\text{Questions}}{27}\right)*100$$

$$TotalScore=\left(\frac{\sum_{1}^{16}\text{Questions}}{144}\right)*100$$

$$S\text{-}RScore=\left(\frac{\sum_{S\text{-}R}\text{Questions}}{36}\right)*100$$

2. Independent Variables

Several different environmental and ship related variables were measured during the data collection aboard SWIFT in order to relate the MSAQ motion sickness scores to the conditions that the scores were recorded in. These relations will enable development of a model that can predict motion sickness scores based on the conditions that personnel will be exposed to.

a. Ship's Speed

An hourly log of ship's speed was maintained on SWIFT's bridge by the Officer of the Deck. This log was kept continuously from departure to arrival except for periods that the ship was import for refueling. Because the speed log is hourly and the times that scores where recorded vary, the scores were correlated to the speeds from the most recent hour. One drawback to the speed data is that during the data collection transit, the speed remained almost constant. This caused variable speed to have little effect in predicting motion sickness scores based in the data from this experiment.

b. Wave Effects

Two different variables were taken into consideration when describing the conditions that the seas impose on a vessel: one is the height of the wave, and the other is the relative direction in which the waves are approaching the ship. The wave height was recorded hourly on SWIFT's bridge in the same log as ship's speed. The recorded heights of the waves were an estimate based on the observations of the bridge watch team. An example of a log entry is that the height of the waves is four to five feet. The larger of the two parameters in each log entry were used since these are the most significant wave heights observed in that time period.

The relative wave direction was also recorded hourly in the bridge log. Relative wave direction is the direction that the waves are traveling from the perspective of an individual facing the bow of the ship. Values range from 000 to 360 with 000/360 describing waves coming at SWIFT head on and 180

describing waves approaching the ship directly from the stern. The reason that relative wave direction is important is that the angle of the wave as it passes under a ship affects the frequency and intensity of the motion that the person onboard is exposed to. In order to effectively relate the relative wave direction with MSAQ scores, the wave direction was partitioned into a three level categorical predictor: head seas coming from the direction of the bow, following seas coming from the direction of the stern, and beam seas coming from the directions of either port or starboard beams (Figure 9).

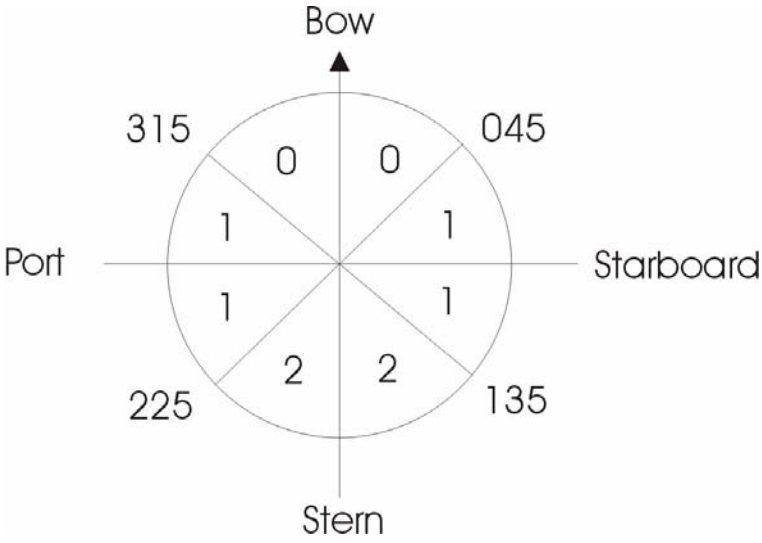

Figure 9. Relative Wave direction Chart

c. Time of Exposure

The time that a subject is exposed to wave motion is another variable that was measured in order to gauge the effects of sensory adaptation to MSAQ scores. Studies of motion sickness have shown that the effects of adaptation to motion sickness are the most significant during the first 48 hours of the exposure, so it is expected that a large portion of the subjects will show signs of adaptation since the exposure times are greater than five hours. The zero

hour of exposure time was when the ship left port. Every observation from the MSAQ surveys was correlated by hours underway from zero hour.

d. T-foil

The design of the HSV-2 SWIFT was modified from previous HSV designs by the addition of a Dynamic Active Ride Control in an effort to reduce the severity of wave effects. This ride control consists in active trim tabs aft, and a retractable T-foil located forward at the end of the center bow (Figure 10). The T-foil is optional in that it may be activated by the bridge watch standers at any time in order to produce a smoother ride through rougher seas. Periods that the T-foil was active were also recorded in the deck log on an hourly basis by the Officer of the Deck.

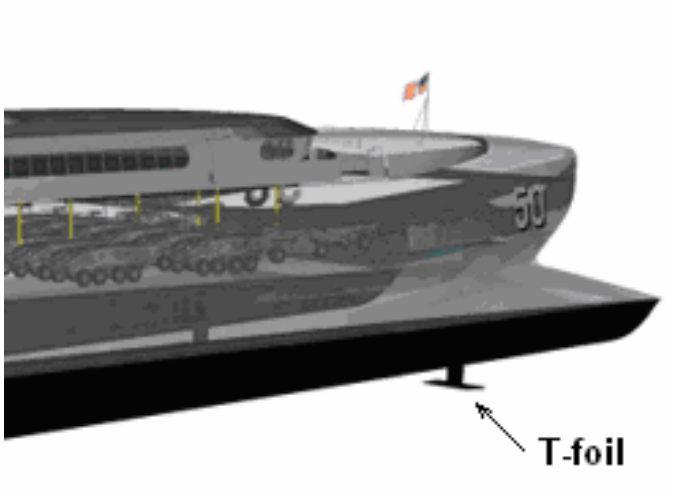

Figure 10. T-Foil Illustration

e. Member of the Crew or Embarked Passenger

Another variable that was recorded as an independent variable was whether the person was a part of the crew or not. The question was considered because a difference might be found in the behavior of people that have adapted to the type of motion that the HSV-2 produced versus those that are not.

3. Model Development

A standard linear regression model was used to determine the relationships between motion sickness scores (MSAQ) on SWIFT and the independent variables that influence motion sickness.

In this model, let y_i represent the random independent variable (MSAQ score) for the i^{th} individual; $x_{1i}, x_{2i}, ..., x_{ki}$ be values of the k independent variable for the i^{th} individual (where categorical variables are coded as indicator variables). Then $y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + ... + \beta_{ki} x_{ki} + \varepsilon_i$, where $\beta_0, \beta_1, ..., \beta_k$ are the unknown parameters and $\varepsilon_i, i = 1, 2, ...n$ are random errors, where the usual modeling assumptions of independence, zero mean, constant variance σ^2 and normality hold.

Two of the data sets were used along with the statistical program S-PLUS 7.0 (© 1998, 2005 Insightful Corp.) to fit a motion sickness model. The Crew and Marine motion sickness data sets from April 2005 were selected because they include data linking the sickness scores and the conditions that the scores occurred in. The initial approach to fitting the model was to combine the two data sets, but because of the large differences in exposure times between the two (Figure 11) and the fact that the Marines where only observed in high sea states (Figure 12), the decision was made to use the Crew data set to generate the model and to use the Marine data set to test it.

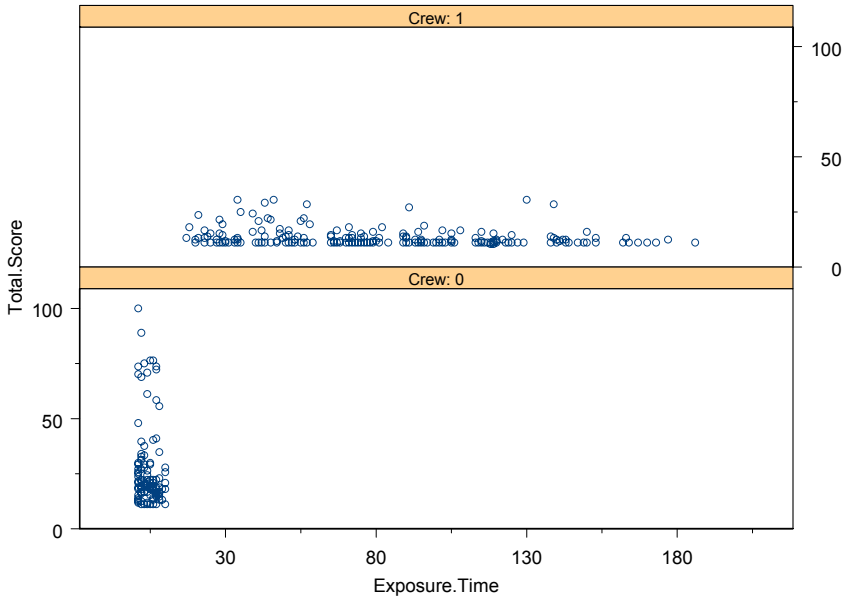

Figure 11. Total Score vs. Exposure Time for Crew (Crew=1) and Marines (Crew=0)

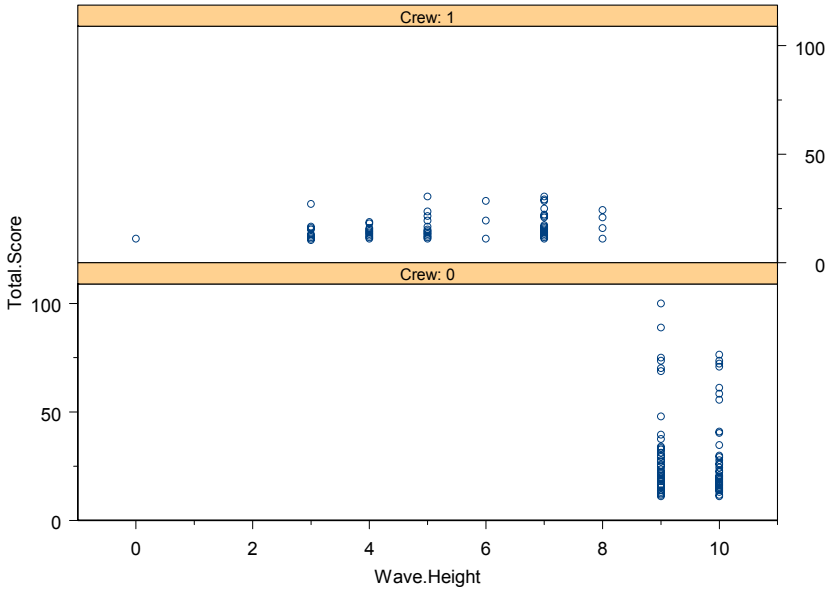

Figure 12. Total Score vs. Wave Height for Crew (Crew=1) and Marines (Crew=0)

This methodology provides an excellent test opportunity since the two data sets were collected independently of each other. The Crew data set was chosen for model development because the wave conditions of the Crew data

set were less severe than those the Marines were exposed to. The resulting model could then serve as a lower bound to predicting the Marine's motion sickness scores.

a. Data Analysis

The first step in model development is to explore the data and check for multicolinearity among the independent variables. To determine if a linear dependence between variables existed, the Variance Inflation Factor (VIF) was calculated for each variable by regressing it against all of the other independent variables (Montgomery, Peck, and Vining, 2001). Acceptable values for the VIF when eliminating the possibilities of multicolinearity are less than 10. In the case of the data set used for model development, the VIF for the independent variables were; Exposure Time: 1.42348, Ship Speed: 1.31614, and Wave Height: 1.21374. These values indicate that linear dependence and multicolinearity among independent predictors were not a problem for this model.

The first linear regression model was fitted from the crew data set with "total sick score" as the dependant variable and "wave height," "ship speed," "exposure time," "wave direction," and all two way interactions between them as the independent variables. The T-foil was used for almost all of the data collection period so this variable was left out of the model formulation. The model yielded a residual plot that shows increasing variance with many residual values close to zero (Figure 13).

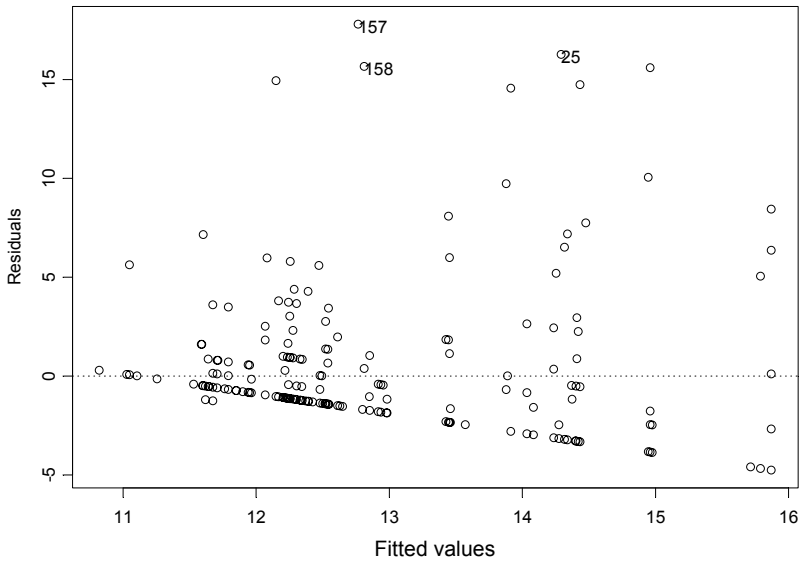

Figure 13. Plot of Residual vs. fitted values for 1st linear Model fit

The MSAQ test score that represents the lowest level of motion sickness is 11.11 which indicate a subject that answered every question with a 1. 189 of 397 observations fall in that condition. To eliminate the number of zero-valued residuals, the data set was sub-setted to include only those scores above 11.5. In other words, the model would only take into account those observations of personnel that had some motion sickness. The new fit using the sub-setted data gave a much better spread in the variance, but there were still signs of heteroscedasicity as can be seen in Figure 14.

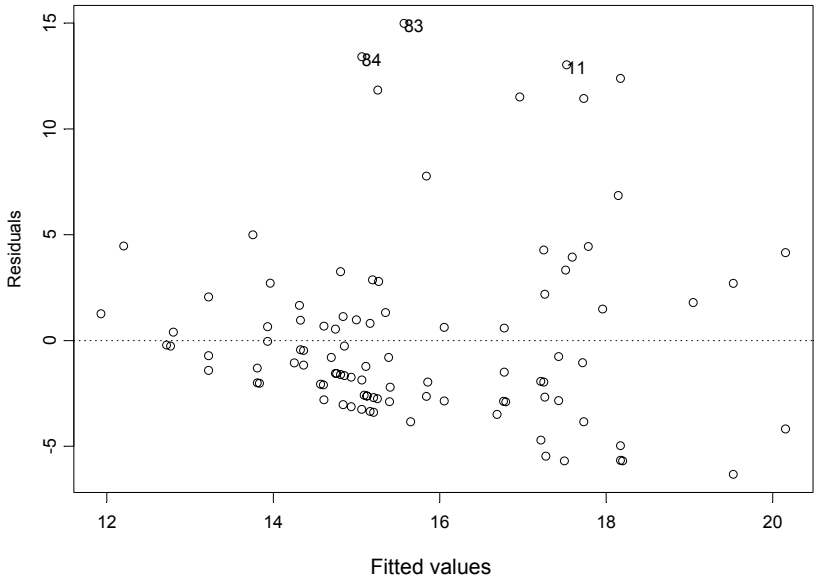

Figure 14. Plot of Residual vs. fitted values for Sub-setted Linear Model fit

Due to the persistent unequal variance in the residual plot, the variance stabilizing transformation $-\left(\frac{1}{TotalScore^2}\right)$ was used. The variance spread for this model was adequate for the equal variance assumption of linear regression models to hold and allows further analysis with this model (Figure 15).

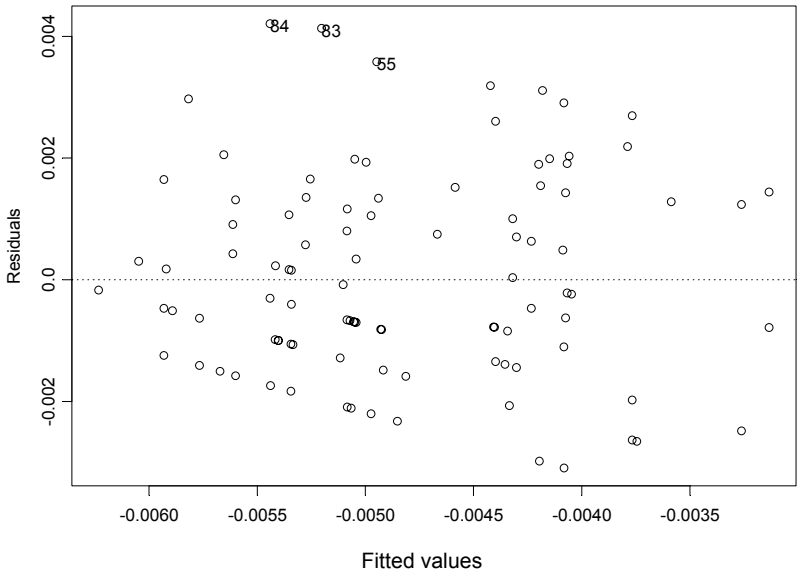

Figure 15. Plot of Residual vs. fitted values for $-\left(\frac{1}{TotalScore^2}\right)$ linear Model fit

In order to analyze the effects each independent variable in the presence of the other predictors, partial residual plots where inspected for patterns in each predictor. Figures 16, 17, 18, and 19 show the partial residual plots for all of the independent variables; no patterns were found. Additionally, inspection of the quantiles of standard normal plot yields that the normality assumption for errors (ε_i) for linear regression models holds true (Figure 20).

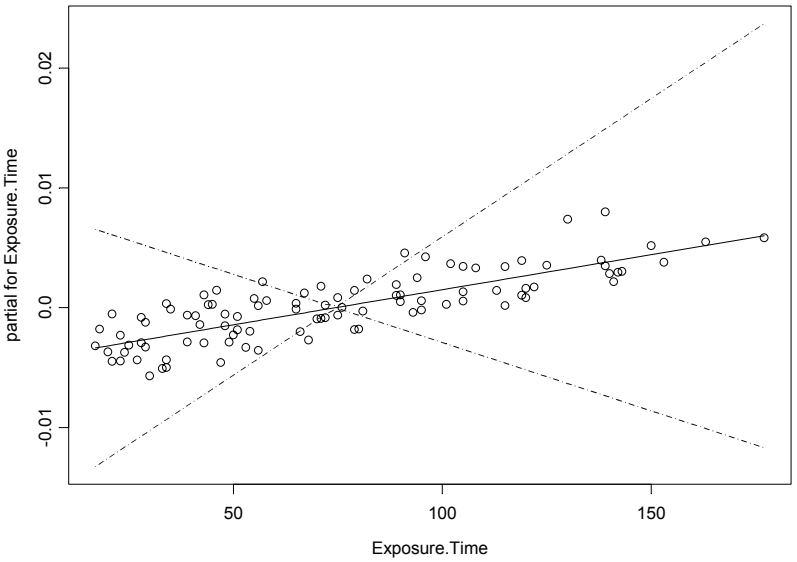

Figure 16. Partial Residual Plot for Exposure Time

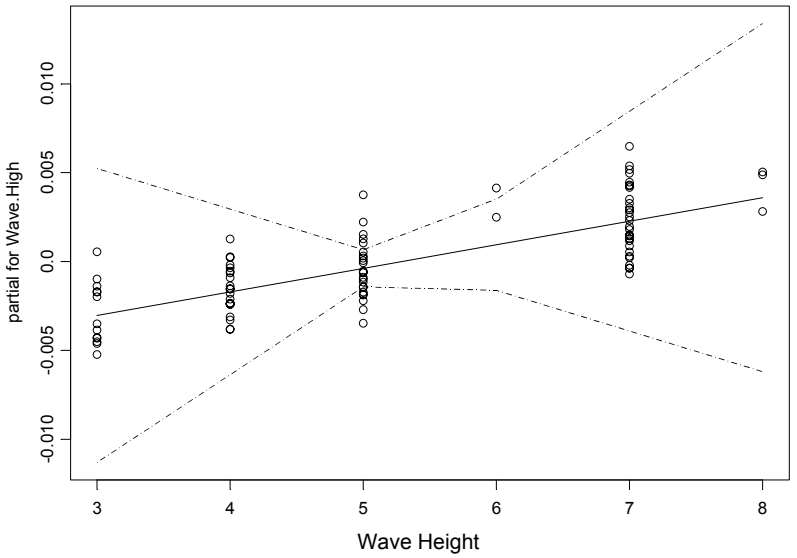

Figure 17. Partial Residual Plot for Wave High

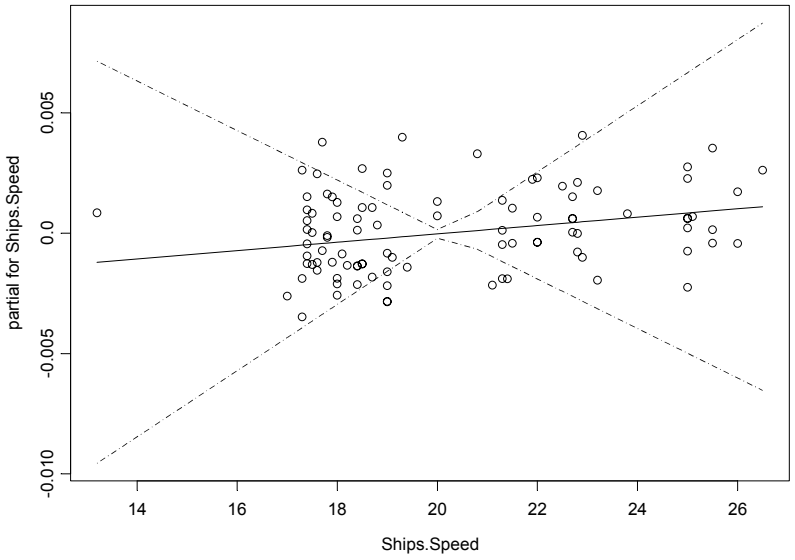

Figure 18. Partial Residual Plot for Ship Speed

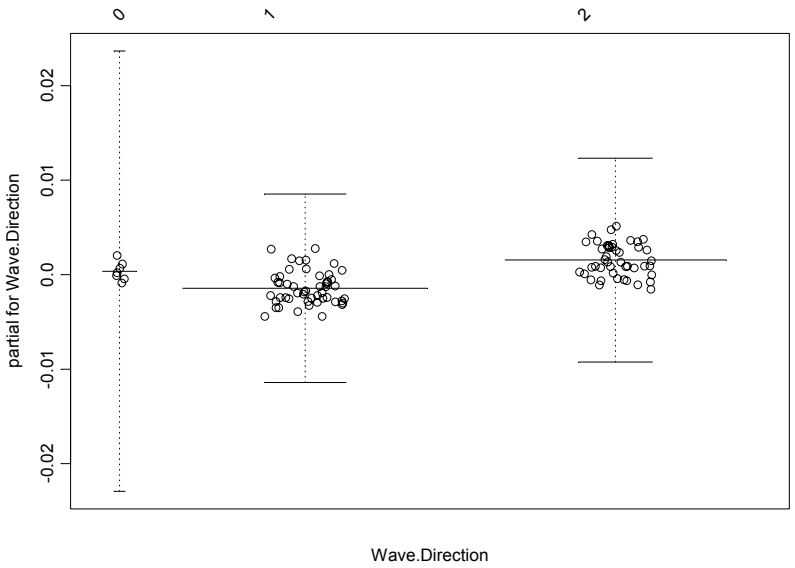

Figure 19. Partial Residual Plot for Wave Direction

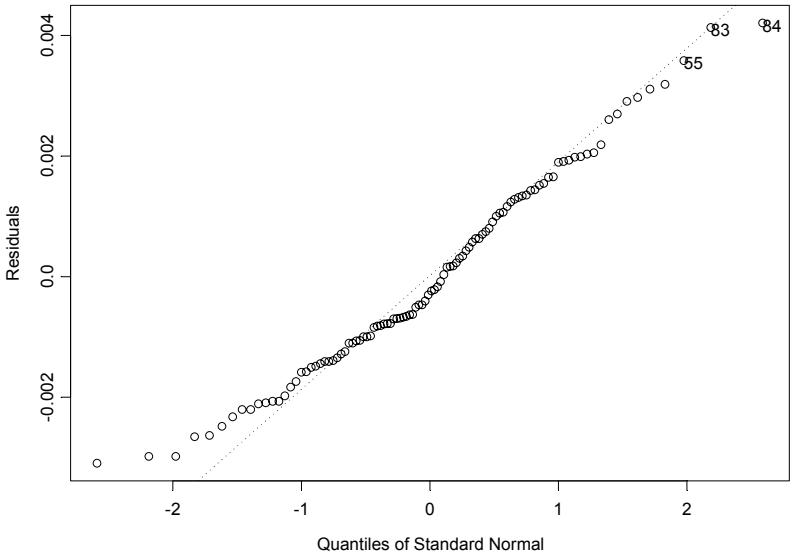

Figure 20. Normal Plot for Residuals

Once the linear regression model assumptions are satisfied, the effects of each of the different parameters and their two way interactions were checked in order to determine if they influence the model. The resulting estimates of β_j, $j = 0,1,...,k$ values for each predictor as well as the associated standard errors and p-values for test of partial effects are displayed in Table 2.

Table 2. Estimated β_j values for the motion sickness model

	Value	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.01450	0.01470	-0.98890	0.32540
Exposure.Time	0.00010	0.00010	0.67890	0.49900
Wave.Height	0.00130	0.00180	0.73460	0.46450
Ships.Speed	0.00020	0.00060	0.28880	0.77340
Wave.Direction1	-0.00090	0.00690	-0.13150	0.89570
Wave.Direction2	0.00070	0.00320	0.21610	0.82940
Exposure.Time:Wave.Height	0.00000	0.00000	-1.73660	0.08590
Exposure.Time:Ships.Speed	0.00000	0.00000	-0.28770	0.77420
Exposure.TimeWave.Direction1	0.00000	0.00000	0.76800	0.44450
Exposure.TimeWave.Direction2	0.00000	0.00000	0.28050	0.77970
Wave.Height:Ships.Speed	0.00000	0.00010	0.06340	0.94960
Wave.Height:Wave.Direction1	-0.00030	0.00040	-0.70580	0.48210

Wave.Height:Wave.Direction2	-0.00020	0.00020	-0.88230	0.38000
Ships.Speed:Wave.Direction1	0.00000	0.00030	0.06810	0.94590
Ships.Speed:Wave.Direction2	0.00000	0.00010	-0.03700	0.97060

It was observed from several high p-values that many of the independent variables are not needed to predict motion sickness scores. Stepwise selection based on Aiaiki's Information Criteria (AIC) was utilized to perform variable selection (Montgomery, Peck, and Vining, 2001). The stepwise procedure allowed for the addition of pair wise interaction terms. Once variable selection was complete, the only variables left in the model where "Exposure Time" and "Wave Height." Table 3 lists the estimated values for the β_j 's in the final model.

Table 3. Estimated β_j values for the Final MS model

	Value	Std. Error	t value	Pr(>|t|)
(Intercept)	-0.0052	0.0008	-6.6145	0.00000
Exposure.Time	-0.00001	0.0000	-2.2434	0.0271
Wave.Height	0.0002	0.0001	1.9283	0.0566

The final linear regression model for motion sickness in SWIFT is $\hat{y} = -52.15961*10^{-4} - 10.2031*10^{-6} * x_1 + 22.25404*10^{-5} * x_2$, where $\hat{y}$ is the estimated expected transformed sick score, x_1 is "Exposure Time" and x_2 is "Wave Height."

The final motion sickness model has an F-statistic of 5.863 (p-value = 0.0039) for the test of the null model with no independent variables against the full model with the independent variables "Exposure Time" and "Wave Height."

Note: All of the detailed motion sickness models and statistical inferences can be found in Appendix B.

C. MOTION INDUCED TASK INTERRUPTION METHODOLOGY

1. Dependent Variable Watch Utilization

While the mathematical models for tipping (Baitis, 1984) predict MIIs with some degree of accuracy, they are limited to those tasks that are interrupted by tipping and require detailed motion data for their inputs. This study will take a more general approach to the analysis of the degradation of task performance due to interruptions due to motion effects than that of the previous researchers in this field. In this study, motion induced task interruptions (MITI) are broadly defined as periods of time that a watch stander is unable to work on their current task because of the motion of the vessel. Watch standers may be seated, standing, or walking and the MITI events may include, but are not limited to, stopping and holding onto a grab rail while walking, bracing against a bulkhead while standing, or holding onto a desk or console while seated.

This general approach does not model the motions of the ship. It takes the conditions that the ship is exposed to and translates that directly into interruptions that personnel experience. The interruptions are categorized by their durations and frequency. These factors are used as input parameters the task simulation model to determine the effects of motion on the task performance of the modeled watch stander. The dependent measure of performance is watch utilization.

Watch utilization is the percentage of total time that a watch stander is busy working on their tasks. This time takes into account the time that is spent holding on or bracing during a MITI. These values are collected at the end of each simulation run and are represented as decimal values between 0.0 and 1.0. Watch utilizations that are closer to 0.0 indicate that the watch stander is capable of completing all assigned tasks. Watch utilizations closer to 1.00 show that the watch stander could be overburdened and may have trouble completing all assigned tasks. By comparing the varying values of watch utilizations in all

modeled watch stations and in all modeled missions, this study may indicate areas of interest in which certain watch standers are susceptible to becoming overburdened due to the occurrence of MITIs.

2. Independent Variables

There are several independent variables taken into account for the study of the effects of MITIs on task performance. These variables were partitioned into two categories: those that have an effect on determining the rates and intensities of MITIs and those that affect the input parameters to the Simkit simulation model.

a. MITIs

The independent variables that allow MITI rate and intensity to be determined include some of those utilized in the analysis of motion sickness. The variables that are the same are: ship's speed, wave effects, and T-foil use. In addition to these, vertical accelerations (z-axis), subject position and camera location were considered.

(1) Vertical Acceleration (z-axis). The z-axis accelerations were recorded by several accelerometers that were placed on SWIFT during a transatlantic voyage. The data from one accelerometer placed near the centerline of the ship in the forward part of the galley was used. The data provided by this accelerometer displays the z-axis acceleration as a percentage of force of gravity in addition to the regular force of gravity at sea level (g's). This data was automatically recorded at time intervals of 30 seconds or less and is linked to the Greenwich Mean Time (GMT or ZULU time) that it was recorded.

(2) Subject Position and Camera Location. The position that the subject was in when they experienced a MITI was recorded as a four level predictor: standing, walking, seated, or kneeling or squatting. These were recorded along with the duration of time that the subject's current task was interrupted as well and the time of the interruption (ZULU). These MITIs were recorded by analysis of video that was taken aboard swift in several selected areas by stationary video cameras during most of the underway period across the Atlantic. The camera location that taped the MITI was also logged with the reset of the data relating to that MITI in order to determine if certain parts of the ship were subject to different severities of MITIs.

b. Simulation Model

The HSV-2 simulation model requires 24 input parameters. 15 of these input parameters are used to model the times that it takes watch standers to complete certain tasks. These time parameters were collected from observed tasks performed on the HSV-2 as well as from interviews of qualified watch standers for those tasks not observed on SWIFT. These 15 task times are:

- Time between engineering rover (ER) rounds.
- Short and long transit/ job times for the engineering rover.
- Time it takes the Navigator of the Watch (NOOW) to update the ship's deck log.
- Time it takes the NOOW or the Officer of the Deck (OOD) to designate a contact on the RADAR display.
- Time it takes the NOOW to update the ships position log.
- Time it takes the OOD/ NOOW to manage/ review the ships position in the electronic charting system.
- Time it takes the OOD to change ship's course utilizing the manual method without autopilot.
- Time it takes the OOD to change the ship's speed by adjusting engine throttle controls.
- Time it takes the OOD to complete a standard bridge to bridge radio call.
- Time it takes the TAO to evaluate a new contact.

- Time it takes the TAO to talk on a radio circuit.
- Time it takes the TAO to review an engagement.
- Time it takes the TAO to monitor the global command and control system.
- Time it takes the TAO to plan a strike mission.

These times are modeled in the simulation by random variables from different Gamma distributions each of which is closely approximated by normal distributions with no negative time values. The other nine input parameters are used to control the simulation in order to model different missions. They include:

- Interval between navigational fixes which varies with the distance the ship is from land.
- Inter arrival times of surface contacts which are varied to simulate different maritime traffic densities.
- Inter arrival times of other contacts which are varied to simulate different warfare missions.
- Time between command initiated course and speed changes which is varied to simulate navigation close to land or warfare maneuvers.
- Time between strike missions which is varied to simulate calls for fire support from forces ashore.
- The probabilities that course and speed changes are required to facilitate safe navigation.
- The probability that a new contact will be engaged which is varied to simulate environments of different hostilities.

The parameters that are probabilities are entered into the simulation as double values. Those that are inter arrival times or times between events are represented by Exponential random number distributions. The values for these parameters are assigned minimum and maximum values which are used along with the MITI parameters in the Nearly Orthogonal Latin Hypercube experiment design process to evaluate their significance to the simulation results.

3. Model Development

a. Task Analysis

The first step to determine the effect of MITIs on important tasks is to determine which tasks are important enough to model. After considering several positions on the SWIFT, it was determined that the positions of the Officer of the Deck, the Navigator of the Watch, and the Engineering Rover were the watch stations that would always be manned and are susceptible to interruptions. Other watch stations that were considered were the Engineering Officer of the Watch (EOOW), the galley crew, the hospital corpsman, and the Command, Control, Communications, Computers, and Information (C4I) watch. The EOOW was not included in the model because of the level of automation on SWIFT. The machinery control systems automatically start, align, and stop generators when required. The task of bringing main engines online is more complex, however, but in any critical situations, all main engines will already be online. The galley crew was not modeled because, while feeding the crew is important, there are always quick alternatives such as meals, ready to eat, and battle rations that can be used if the sea states are too high to operate the galley. The hospital corpsman was left out of the model because if there was a critical situation in which treatment would be required for a life threatening situation, it would most likely wait until the ship could be slowed or steered to avoid a high level of MITIs. The C4I watch was not modeled because of the need to model a combat watch stander such as the TAO. The TAO model in the simulation is burdened with most of the tasks a typical C4I watch station has as well as those required of a more senior watch station. The TAO model should thus suffice in this model to simulate the typical combat watch stander.

A hierarchical task analysis (Wickens, Lee, Liu, and Becker, 2004) was then conducted of the three watches aboard SWIFT that were chosen to be modeled. This was done by recording the different tasks associated with each watch, the stimulus for each task, and the processes that are required to

complete each task. A hierarchical task analysis was also conducted of a TAO watch by interviewing 10 Surface Warfare Officers who were qualified TAOs aboard CG and DDG class ships. These task analyses were recorded and translated into JAVA to construct the Simkit simulation model that was used for the analysis of MITIs.

b. Simkit Model

The simulation model used to model tasks shipboard tasks in this study was constructed in Simkit, a JAVA based discrete event simulation package that works with any JAVA integrated development environment (Buss, 2002). Discrete event simulation models can be graphically represented by event graphs (Buss, 2001). Event graphs represent the future event list logic components that make up discrete event simulation: parameters, state variables, and the event list. Parameters are the values that are input or initialized at the beginning of the simulation run. Some examples of parameters are the probability that a course change is required or the set of random times between events. State variables are used to keep track of the progress and change with time as the simulation runs. Examples may include the total number of interruptions, a queue, or a variable that keeps track of whether or not a simulation entity is busy. The final component is the event list which contains the list of future events (or tasks) that have been scheduled to occur as well as the time that they will be executed. The execution of the events takes no simulation time, but the follow-on events are scheduled with a time delay to represent the amount of time that it takes to complete an event. These events are sorted in the event list in sequential time order and are executed one at a time by the event with the soonest scheduled execution time. This is what categorizes these simulations as "event step." The simulation time does not flow from zero to the simulation end time, but jumps to the time that the next event occurs.

Event graphs are composed of nodes that represent events with associated state variable transitions and arcs or "scheduling edges" that represent the process of the source event adding the destination event to the

event list for execution. Scheduling edges may include delay times, which are usually either constant or drawn from a set of random variates, or edge conditions that only allow the destination event to be scheduled if the logical conditions are met. There are also “cancelling edges” represented by dashed lines that remove events from the event list. These constructs give event graphs the capability to represent almost any system with a certain degree of abstraction.

The simulation model developed in this study consists of seven simulation entities, each modeled in a separate JAVA class, as well as one assembly class to initialize and execute the simulation. Of the seven simulation entities, four are used to model shipboard watch standers and three are used as controls to model different conditions that the watch standers are exposed to. Varying these conditions allows the simulation to model different missions and degrees of MITI frequency and intensity that the watch standers may be exposed to. The watch entities also react to each other in the same hierarchical manner as watch standers would on the SWIFT or may aboard a LCS class ship.

The assembly (Buss and Sanchez, 2002) program, HSV-2 Model Assembly (see APPENDIX E) controls the execution of the simulation runs. It reads parameters from an input file of comma-separated values and uses these parameters to initialize the simulation entities as well as the simulation run parameters. It then constructs the simulation entity listening pattern to implement the hierarchy of watch stations and allow the three control entities to affect the four watch stander entities in the appropriate fashions (Figure 21). It also constructs the data collection objects that are used for gathering statistics after each simulation run. The assembly class then executes the simulation for the desired time period and prints the gathered statistics to an output file that can be opened in a spreadsheet program and imported into a statistics tool such as S-PLUS© to facilitate experimental analysis on the simulation output.

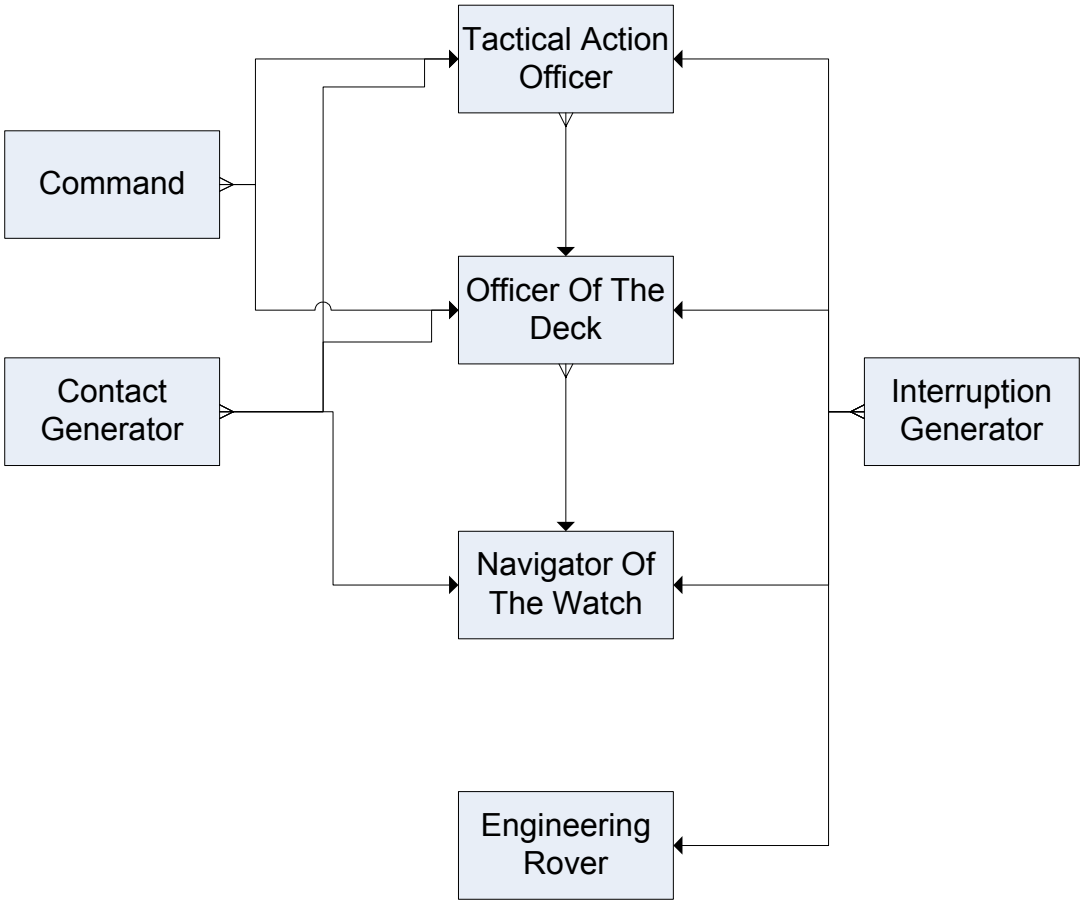

Figure 21. HSV-2 Model Assembly Sim Event Listener Pattern

The control class that implements MITIs in the simulation is the Interruption Generator (see APPENDIX E). The Interruption Generator starts a period of interruption after a randomly generated time interval from a specific distribution. The Start Interrupt event is heard by the listening watch stander classes and causes each to stop work on their current tasks. After a randomly generated interruption length from a separate distribution, the Interruption Generator fires an End Interrupt event that is heard by the watch stander classes that then resume the tasks that they were working on before the interruption occurred. The Interruption Generator then schedules another interruption start event after another randomly generated time interval. The process continues to cycle in this manner until the simulation is terminated. The input parameters for

the random variable generators for interruption rates and durations are those taken from the HSV-2 data collection during the transatlantic crossing.

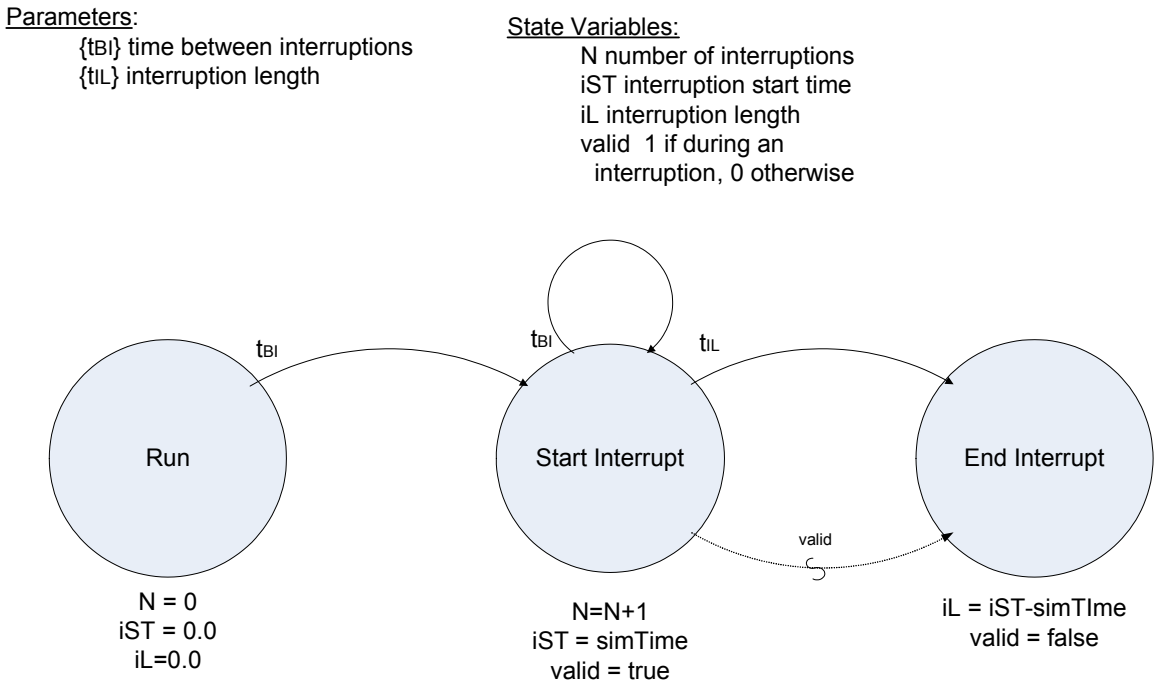

Figure 22. Interruption Generator Graph

The next control class is the Contact Generator (see APPENDIX E). The Contact Generator schedules the arrivals of new contacts that must be processed by the various watch standers as they arrive. These contacts are divided into two groups, surface contacts and other contacts. The times between the arrivals of these contacts, inter-arrival times, are drawn from separate random variable distributions. These two contact types are treated differently by the watch stander entities: surface contacts are contacts that the Officer of the Deck and Navigator of the Watch monitor and may have to maneuver with some probability to avoid, other contacts are those that the Tactical Action Officer monitors and may have to engage when an associated warfare mission is modeled. These contacts are not separated for an anti surface warfare and high speed transit missions since they are the same surface contacts that the Officer

of the Deck and Tactical Action Officer must respond to. However, for other missions that may eventually be of interest, such as anti submarine warfare or air defense missions, these contacts may be differentiated and will have different randomly generated rates of arrivals.

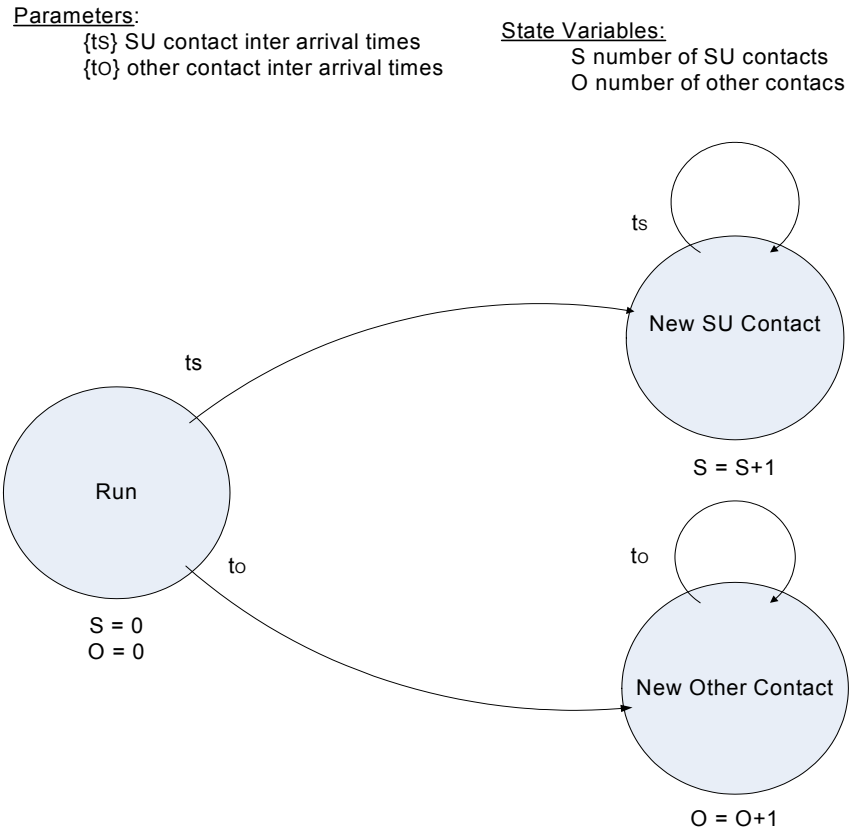

Figure 23. Contact Generator Graph

The Command class (see APPENDIX E) is the class that acts loosely as the commanding officer of a ship. The class schedules course and speed changes with randomly generated inter-arrival times that trigger a course or speed change event in the OOD. These course and speed changes may be used to simulate a mission that may require a high degree of maneuvering such as an ASUW mission in the littorals or high speed transit in the littorals. The class also schedules the occurrence of strike missions with random inter-arrival

times which trigger the TAO to plan and evaluate engagements. This is the method used to simulate naval surface fire support missions and can be used to simulate other strike missions if the need exists. The parameters for the inter arrival time distributions for all of these events are simulation inputs that may be varied to simulate different missions.

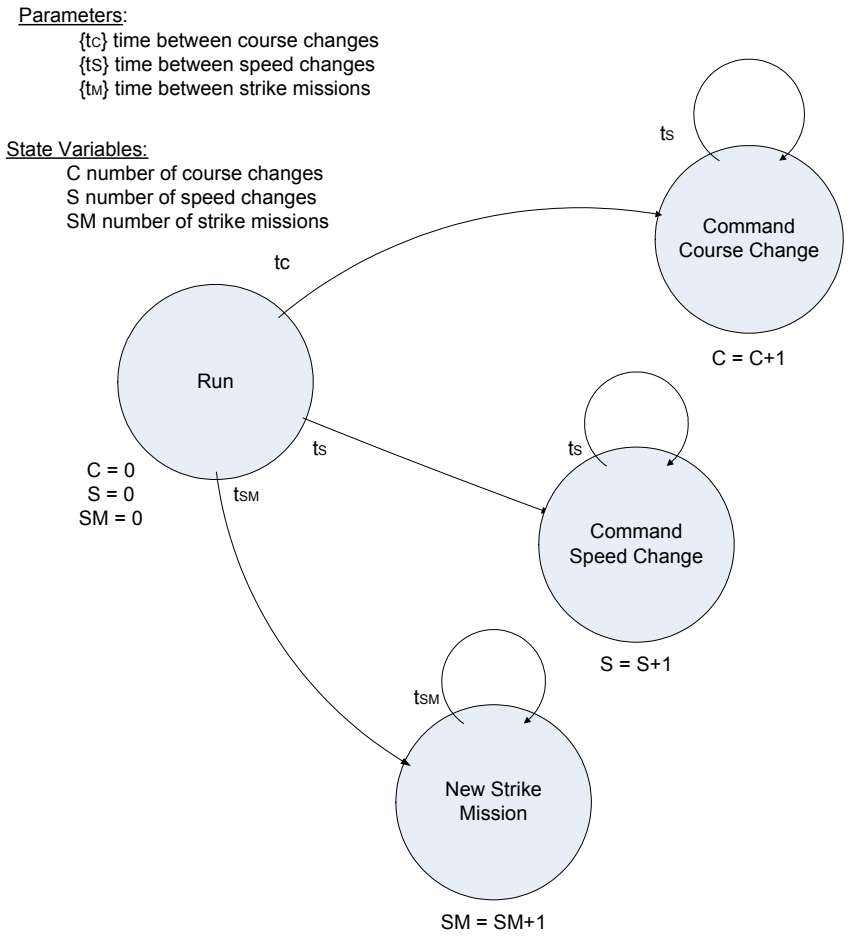

Figure 24. Command Event Graph

The Engineering Rover class (see APPENDIX E) is the simplest watch stander entity modeled in the simulation. The rover entity starts a new round through engineering spaces after a randomly generated time between rounds. The rounds take the rover though all of the engineering spaces on the

SWIFT that the actual watch stander would tour as recorded in the hierarchical task analysis (see APPENDIX D). These spaces include the T-foil void, void #3, port jet room, port engine room, starboard jet room, starboard engine room, and the port and starboard fuel header inspection accesses. The rover entity draws randomly generated times from two random variable generators whose inputs parameters were taken from data collected aboard the SWIFT to simulate the transits to and from, as well as the inspections of the spaces in each round.

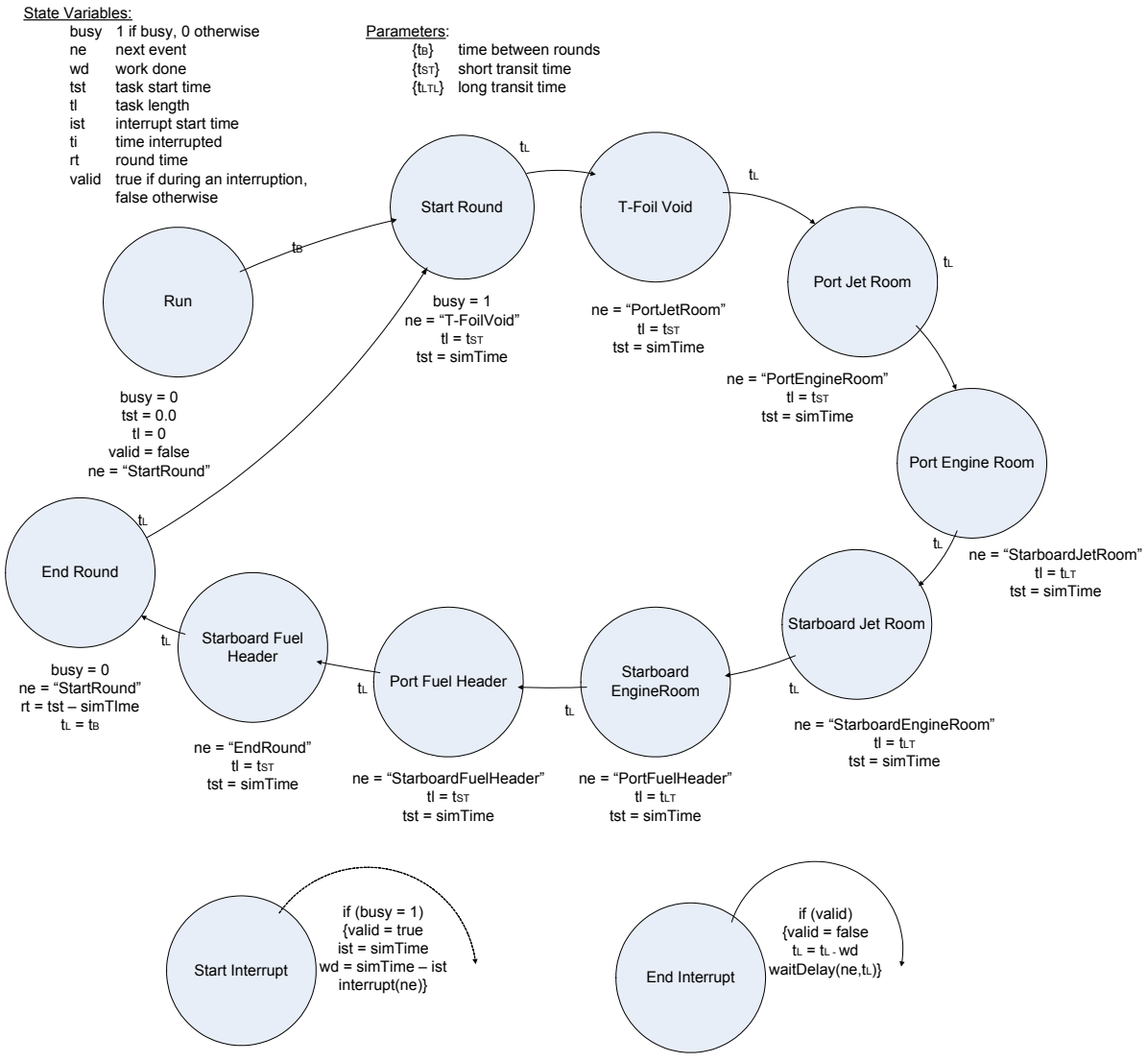

Figure 25. Engineering Rover Event Graph

The Navigator of the Watch class (see APPENDIX E) is the class that simulates the activities of that bridge watch station. The NOOW is subordinate to the Officer of the Deck. The OOD's course and speed change actions cause the watch to check the ship's position and then record the changes in heading and/or speed in the ship's deck log. This is done to model the act of checking the charted water in the new ship's heading to ensure that it is free of navigational hazards and to record the new course and/or speed as done in standard operating procedures. The NOOW also notes the arrival of new surface contacts and plots them on the Automated RADAR Plotting Aid (ARPA). In addition to these tasks, the NOOW also checks the ship's position on the Electronic Chart Display and Information System (ECDIS) and maintains the position log at every fix interval which can range from 3 to 60 minutes depending on the mission the model is simulating. The times that it takes to complete each of these tasks are drawn from separate random variable generators whose input parameters were taken from data collected on the SWIFT.

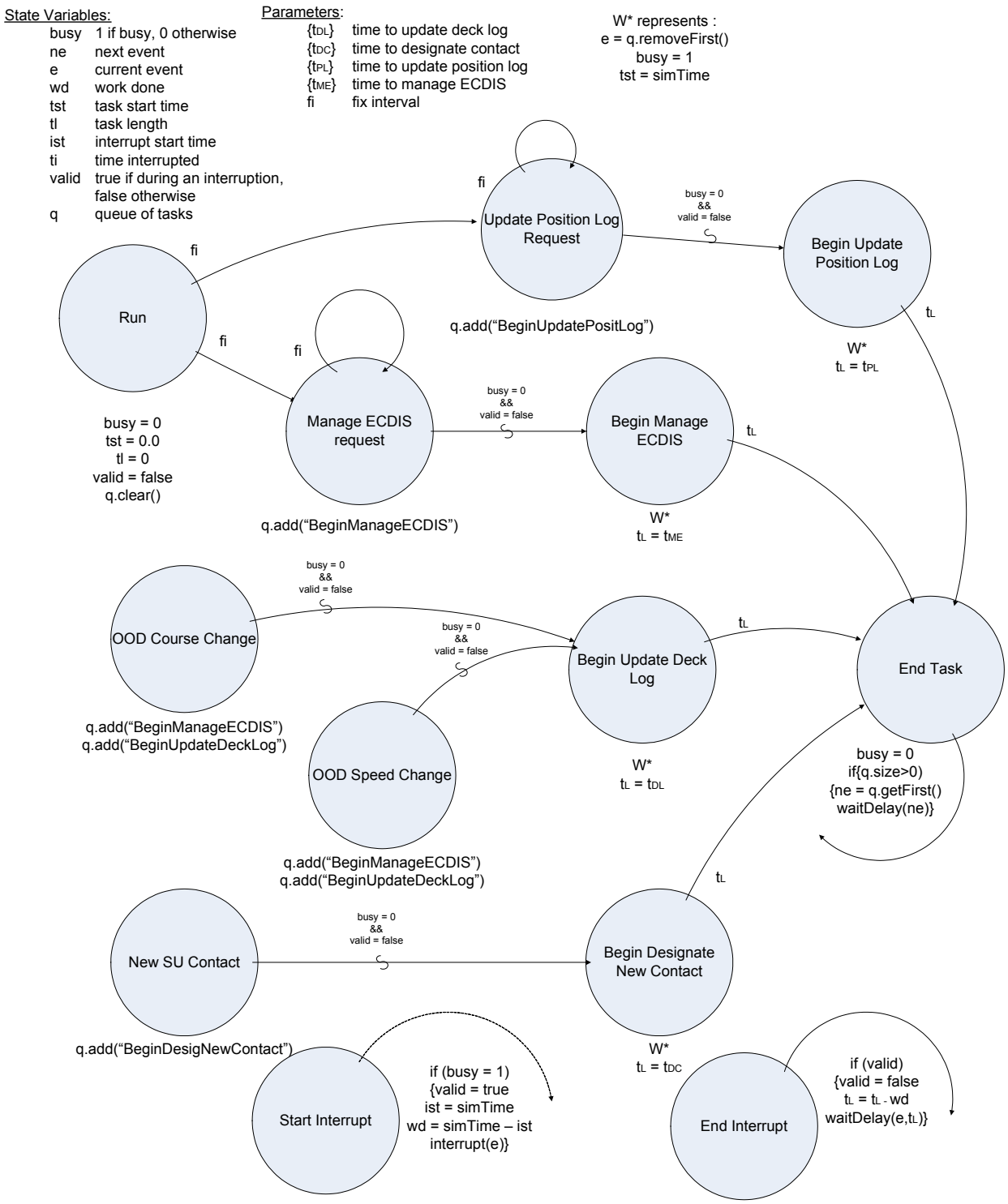

Figure 26. Navigator of the Watch Event Graph

The Officer of the Deck class (see APPENDIX E) models the actions of the primary bridge watch stander on SWIFT. The OOD is subordinate to the Tactical Action Officer and listens to the Command class. Both of these classes signal course and speed change events that trigger the OOD to do the same. The OOD also listens to the Contact Generator class for the arrivals of new surface contacts. These contacts cause the OOD to plot them on the ARPA and, with some probability, change course and or speed for close CPA avoidance. The OOD will also use the bridge-to-bridge radio in the event that course and speed are changed. In addition to these tasks, the OOD will monitor the ship's position on the electronic chart display and information system ECDIS at every fix interval as an actual watch stander would do during an underway watch. The input parameters for the probabilities that course and speed changes are required were taken from interviewing 15 qualified Surface Warfare Officers and vary from 0.0 to 1.0. These are changed for the purpose of simulating different missions and may be varied for sensitivity analysis. The times that it takes to complete each of these tasks are drawn from separate random variable generators whose input parameters were taken from data collected on the SWIFT.

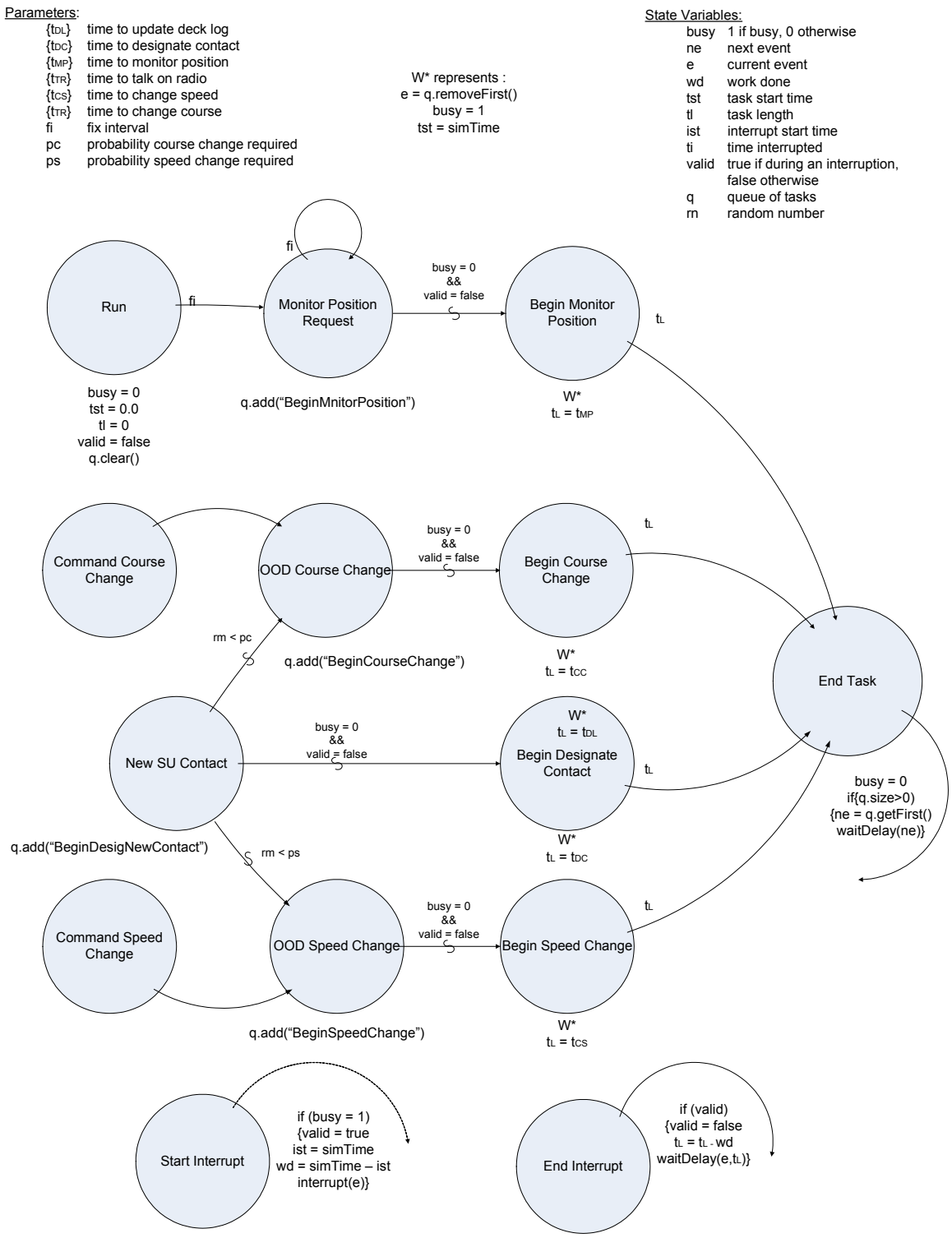

Figure 27. Officer Of The Deck Event Graph

The Tactical Action Officer class (see APPENDIX E) models the senior watch stander in this simulation. The TAO listens to the command class for the arrivals of new strike missions and to the Contact Generator for the arrivals of new other or surface contacts depending on the mission being simulated. The TAO will evaluate each new contact and during warfare missions, will engage that contact with a certain probability. This probability ranges from 0.0 to 1.0 and may be varied for sensitivity analysis. In the event that an engagement is triggered, the TAO will signal a course and speed change to the OOD, initiate a radio transmission event, and then review the associated engagement. The engagements being modeled are gun engagements or quick fire Harpoon type engagements. When a strike mission event is heard for the simulation of NSFS missions, the TAO will complete the same steps as for a surface engagement except for the review engagement event that is replaced by a different review engagement event due to different task completion times. In addition to completing engagements, the TAO also monitors the Global Command and Control System (GCCS) at every fix interval and monitors the communications circuits at randomly generated intervals to simulate communications with other ship or shore commands. Since the SWIFT did not have a TAO on board during the data collection trip, the tasks that are modeled as well as the input parameters for the distributions that control the times to complete each task were taken from interviewing 10 TAOs who were qualified on DDG or CG class ships.

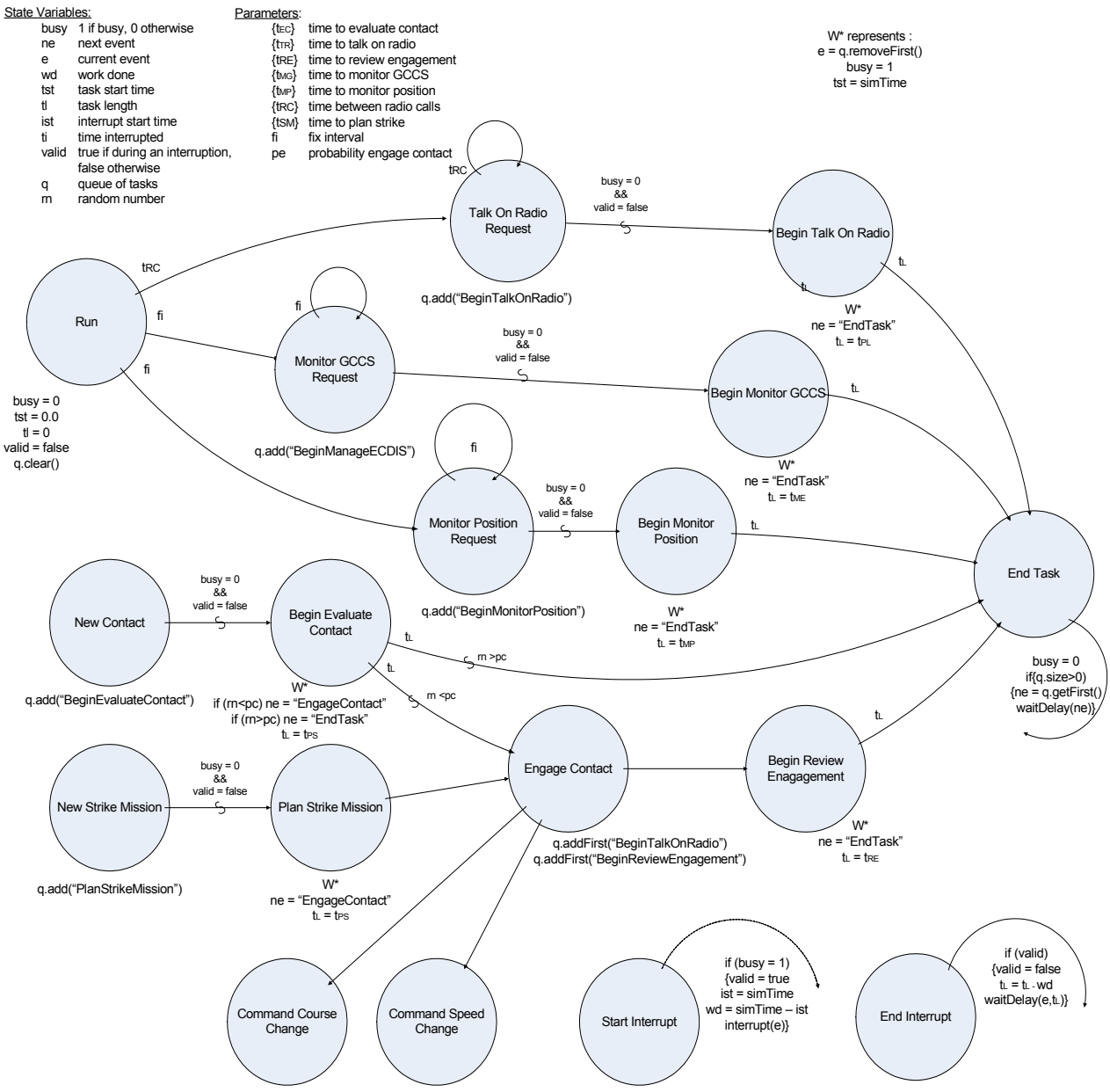

Figure 28. Tactical Action Officer Event Graph

Each of the modeled watch standers has a state variable that keeps track of whether that watch stander is busy or not. This variable is either 0 if the watch is not currently tasked or 1 if it is. If a watch stander entity is interrupted while attempting to complete a task, the "busy" state variable remains 1 until the interruption ends and the task is completed. These "busy" state

variables are tracked by a time varying statistics object that is part of the Simkit statistics package. At the end of each simulation run, the average value, variance, and standard deviation of each watch stander's associated state variable is computed to determine the utilization of that watch stander. Utilization values near 0 indicate a watch stander that had no difficulties completing assigned tasks. Values near 1 indicate a watch stander that may have been overburdened and had difficulties completing all assigned tasks. These utilization parameters are collected for each watch stander in every simulation run and stored in an output file to facilitate ease of analysis.

c. MITI Input Parameters for Simkit Model

In addition to the previously mentioned input parameters for the Simkit model, it was necessary to provide inputs for the length and frequency of task interruptions to determine their effects on performance. The inputs were determined by the application of linear regression models to the MITI data collected aboard SWIFT in April 2005. From these models, prediction intervals were manufactured to select the ranges of MITI Length and MITI frequency to use in the experiment design.

(1) MITI Length. The lengths of 121 observations of interruptions caused by ship motion were recorded in the MITI data sets. These were associated by time with correlating z-axis acceleration, ship speed, wave high, relative wave direction and subject's position prior to the interruption. Since this is the only data set, it was partition into two to provide a training set to fit the model and a test set to validate the model. 21 observations were randomly selected as the test set and were not used in the development of the model; the rest were designated the training set.

A linear regression was fit to the training set to obtain the residual plot and variance spread in Figure 29.

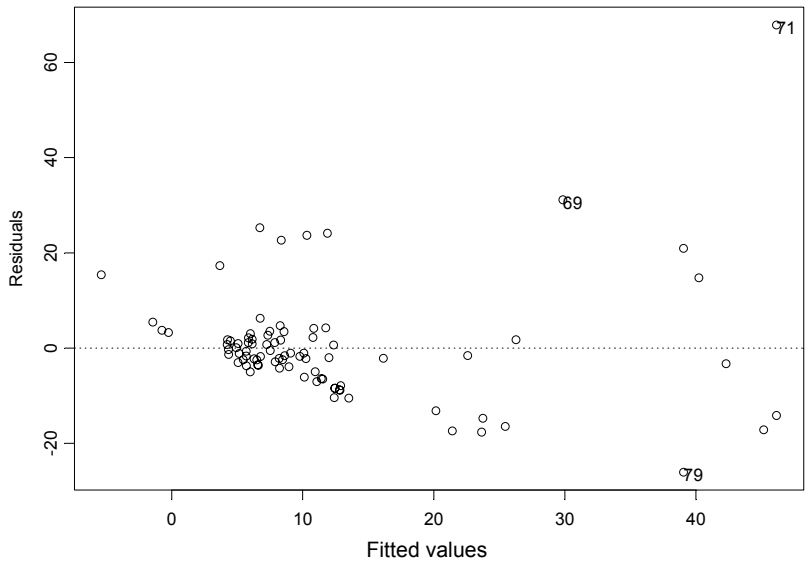

Figure 29. Plot of Residual vs. fitted values for MITI length model fit

Due to heteroscedasicity in the residual plot, a natural logarithm variance stabilization transformation was used. The residual plot for the resulting model meets well enough the equal variance assumption for linear regression models and is displayed in Figure 30.

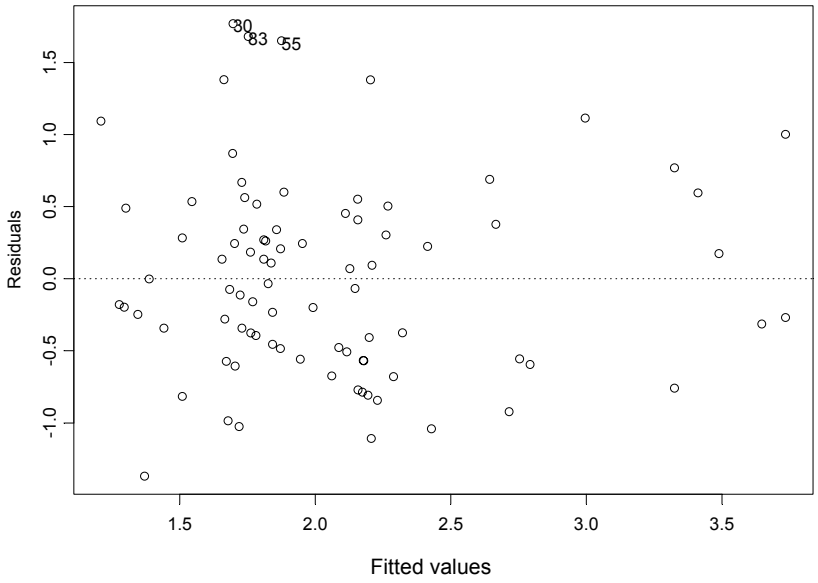

Figure 30. Plot of Residual vs. fitted values for the MITI ln(length) model fit

Stepwise regression based on the AIC criteria was then used to obtain the final model for MITI length (Detailed model can be found in APPENDIX_B):

$$\hat{y} = 8.0784 - 2.0595 * x_1 - 0.8967 * x_2 + 0.3005 * x_3 - 0.3184 * x_4 - 0.2264 * x_5 + 0.2586 * x_6 - 0.0371 * x_7$$

where $\hat{y}$ is the estimated expected natural logarithm of MITI length, x_1 is vertical acceleration, x_2 is "Wave Height," x_3 is "Wave Direction1," x_4 is "Wave Direction2," x_5 is "Position1," x_6 is "Position1," and x_7 is "Position1."

A 95% prediction interval was computed using the final model for MITI length. To verify the model, the test set was superimposed on the plot to determine if most of the test data would fall inside the prediction interval. Figure 31 shows the PI and the test set data, plotted as a function of Wave height and z-axis acceleration for each of the three positions.

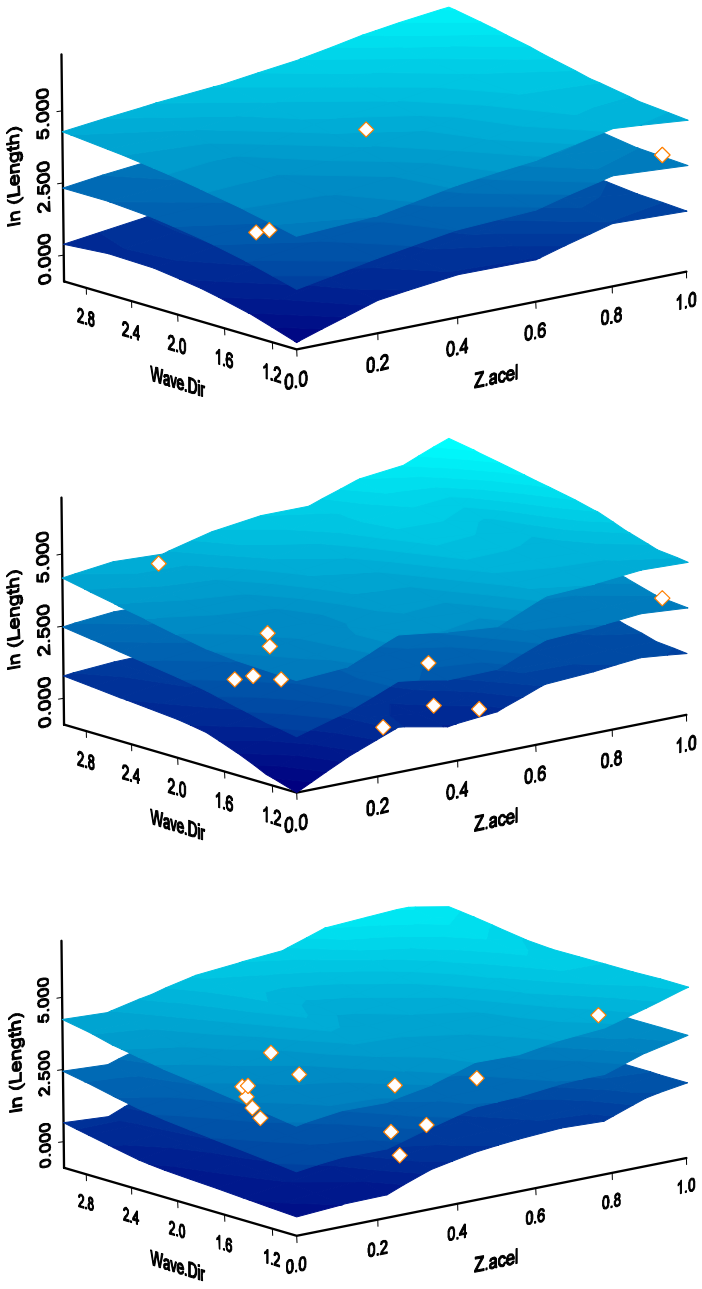

Figure 31. PI for MITI model and test data

It can be observed from the prediction interval plots that the test data is almost entirely inside the PI verifying that the model is an adequate predictor for the length of interruptions based on the data collected.

(2) MITI frequency. Using the MITI data sets from April 2005, every interruption recorded by the cameras was correlated with the vertical acceleration that caused it. The mean value of all of the z-axis accelerations associated with known interruptions was used as the cut off point to determine which z-axis accelerations in the entire data set would cause interruptions. This allowed each z-axis acceleration recorded to be counted either as an interruption event or as a non interruption event. The data was broken into one hour time units and the number of interruptions in each time unit was recorded. The resulting data set was over 100 observations of MITI frequency correlated with relative wave direction, wave height and ship's speed.

A linear regression model was fitted to the interruptions per hour data in order to determine a prediction interval to manufacture simulation inputs. The initial model's residual plot shows an increasing pattern in the variance (Figure 32), so a stabilizing transformation was used.

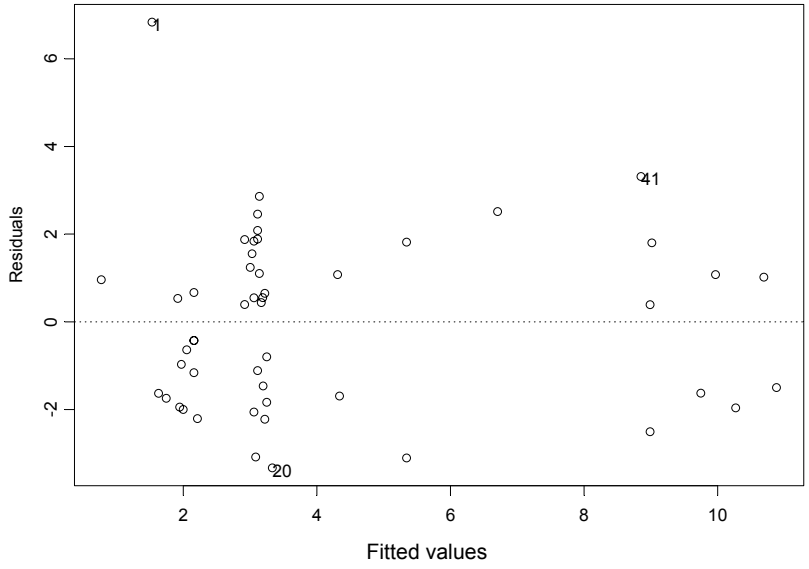

Figure 32. MITI Frequency model variance spread

To stabilize the variance, $\sqrt{(MITI\ frequency)}$ was used to eliminate heteroscedasicity in the model (Figure 33). Stepwise selection AIC was then used to eliminate the parameters not necessary for the model. The

resulting final model for predicting the frequency of MITIs is: $\hat{y} = 12.3967 - 1.0845 * x_1 - 3.2836 * x_2 - 1.6228 * x_3$, where $\hat{y}$ is the estimated expected square root of MITI frequency, x_1 is "Wave Height", x_2 is "Wave Direction1" and x_3 is "Wave Direction2" (Detailed model and statistical inferences for MITI frequency model can be found in APPENDIX B.)

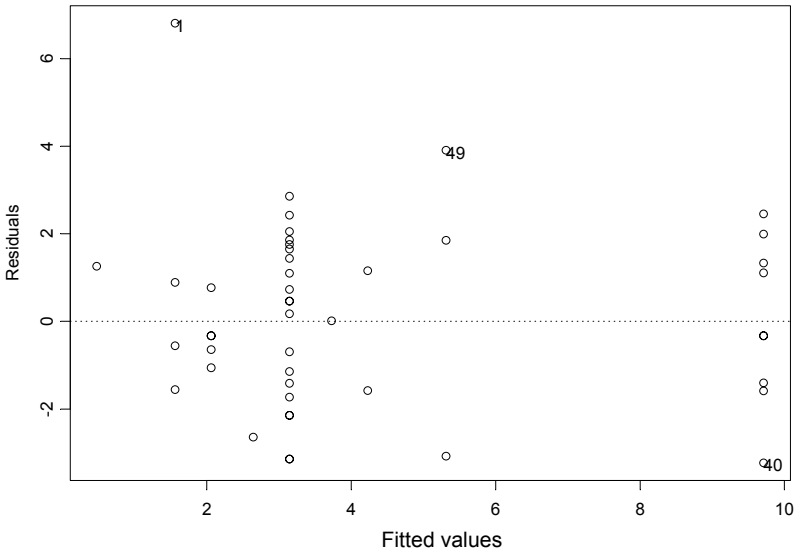

Figure 33. Plot of Residuals vs. fitted values for MITI Frequency model

A prediction interval PI was constructed using the frequency model to allow for the prediction of frequencies that will serve as inputs to the simulation model. Figures 34 and 35 show the PI. In the narrow areas of the plot in which the data was collected, the prediction interval seems to show realistic expectations for MITI frequencies. This is not the case; however, when attempts are made at predictions in areas of the plot away from the collected data. An example of this extremely large variance is shown at wave heights of just over 0.0 feet that are from a direction of 1. The interval predicts MITI frequencies of as much as 360^2 interruptions in an hour. That equates to an interruption every 1.7 seconds which is an unrealistic approximation and yielded

to an alternate method of selecting MITI frequency inputs discussed further in experiment design.

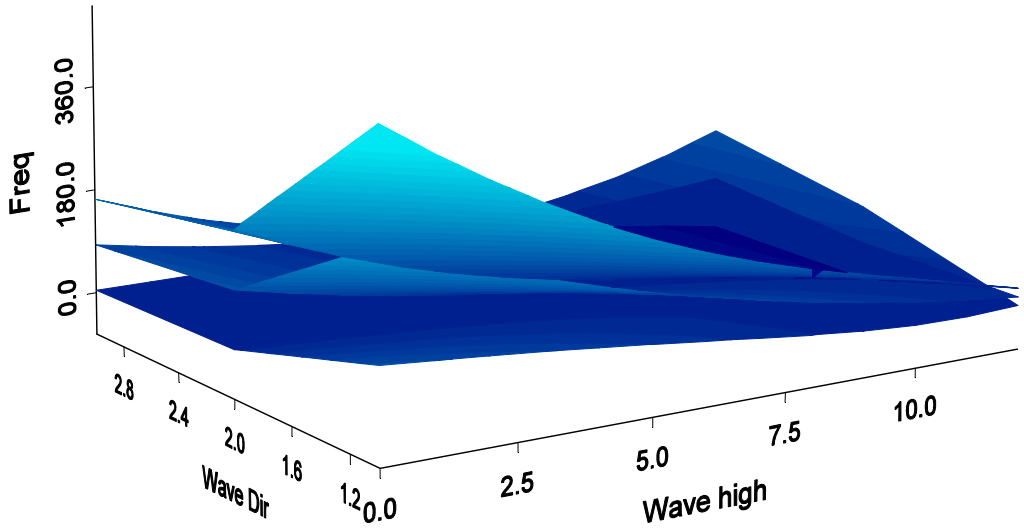

Figure 34. PI for MITI frequency

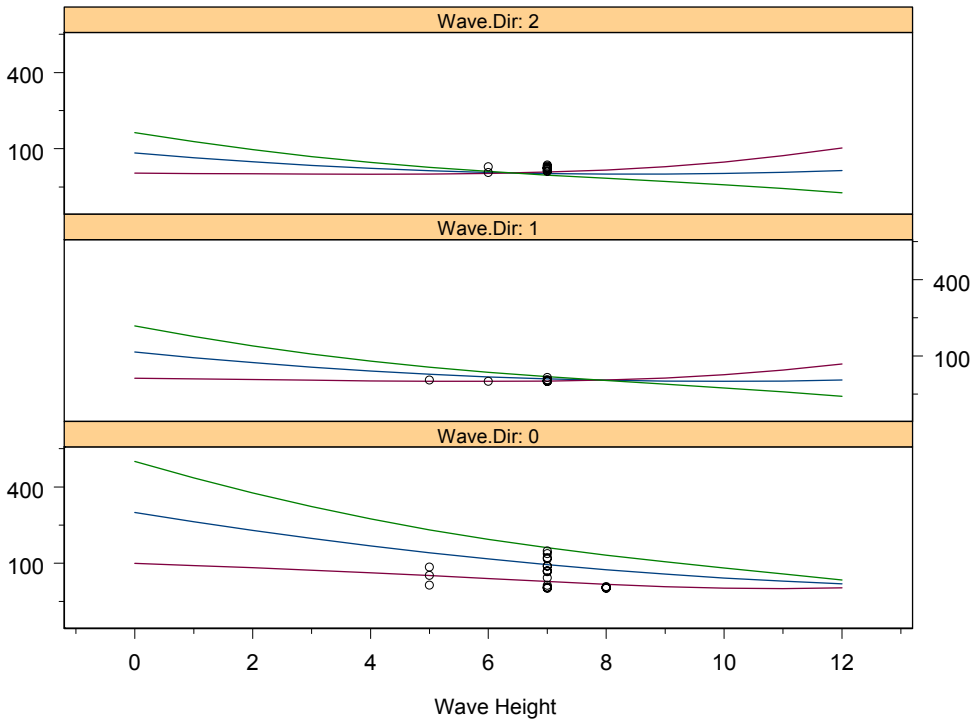

Figure 35. Plots of Prediction Intervals and estimated expected frequency vs. Wave Height for Wave Directions 0, 1 and 2.

d. Experiment Design

A Nearly Orthogonal Latin Hypercube (NOLH) design was used to arrange the simulation control and MITI input parameters to facilitate an efficient and effective simulation experiment (Sanchez, 2005). The NOLH design generates randomly permutated values, 65 in this case, for each input parameter between the minimum and maximum levels of that parameter. The NOLH design allows experimental designs to be constructed that test multiple independent variables, each with multiple levels, with a relatively small number of simulation runs. It also allows compensation for uncertainty in independent variables by allowing them to fluctuate through estimated minimum and maximum values. This trait was exploited in the modeling of MITI frequency due to the exponentially increasing variances in the prediction interval.

To compensate for the large overestimations in the MITI frequency model, the mean from all of the frequency data of 29.1 and standard deviation of 39.3 were used to compute minimum and maximum levels for the NOLH design. One standard deviation was taken from the mean to yield a maximum frequency of 68.4. The minimum frequency was taken to be 1 in order to provide a full range of levels for the NOLH design. The resulting permutations of this parameter were modeled as Poisson arrival rates with Exponential inter arrival times in the simulation model.

Evaluation of the MITI length model showed that the lengths of interruptions depended on the position of the subject prior to the interruption. Since the engineering rover is a walking watch, and the other watch standers are sitting, they will have to be modeled separately. The minimum and maximum MITI lengths for the ER were 2.1 seconds and 24.3 seconds respectively. These were modeled in the simulation with a Gamma distribution. For the other watches, the ranges for MITI length varied from 5.7 to 66.1 seconds. These were also modeled in the simulation but with a different Gamma distribution in separate simulation runs.

The difference in MITI lengths between watch standers increased the number of runs in the experimental design from four to five. The difference is only one and not four runs because the only parameters that affect the engineering rover are the MITI lengths, frequencies, and the times that are specific to the ER tasks. Since these do not change between missions and the ER does not listen to the other controlling classes, only one run is necessary to simulate the ER tasks. The five simulation runs are: one run to collect data for the engineering rover, and one run in each of the four mission areas (high speed open ocean transit, high speed littoral transit, littoral ASUW, and littoral NSFS) to collect data for the NOOW, OOD, and TAO. The constant simulation parameters and the NOLH design inputs for all experiments can be found in APPENDIX F.

The 65 sets of parameters for each modeled mission, or watch stander in the case of the Engineering Rover, were run for 10 repetitions of 40,000 minutes each. The first 20,000 minutes of collected data from each repetition was discarded to prevent any bias to the simulation output due to the simulation warm up time required for dependent variables to reach steady state values. The resulting NOLH independent variable permutations used for simulation input can be found in APPENDIX F.

III. ANALYSIS

A. MOTION SICKNESS ANALYSIS

In Chapter II, Section B, a prediction model was fit relating motion sickness to exposure time and wave height based on the data collected from the crew of the HSV-2 (Figure 22). Recall that two sets of data where available: one from the crew and one from Marines. The Marine data set was designated a test set for two reasons: (1) the time of exposure was only 10 hours and at high sea states versus the 7 days and a variety of sea states of the other data set; and (2) the Marine data provided an opportunity to test the prediction capabilities of the model against an independent data set in order to validate the model.

1. Prediction Interval

Using the motion sickness prediction model from Chapter II, a 95% Prediction Interval (PI) was computed. The range for the PI was across exposure times between 0 and 24 hours in 15 min intervals for wave heights of 9 and 10 feet. Given that more than 120 observations were used to fit the model, a Normal prediction interval was calculated with upper and lower limits: $\hat{Y} \pm 1.96 * \sqrt{\hat{\sigma}^2 + se(\hat{Y})^2}$, where $\hat{\sigma}^2$ is the estimate variance of the regression in the model, and $se(\hat{Y})$ is the standard error of that particular fit. The test data set (Marines Data) was then plotted against this prediction interval to access the validity of the prediction model.

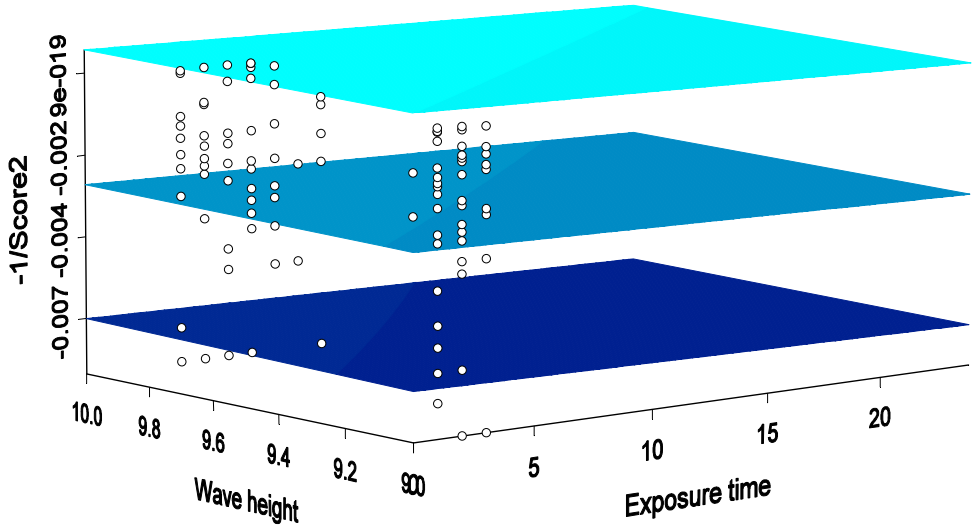

Figure 36. PI and Marines Observations

As shown in Figure 36, only 12 of 117 observations from the Marines data set fell outside the PI. It also appears that the fitted values tend to underestimate the expected scores for the Marines so the model can be used as a lower bound for predicting motion sickness on embarked personnel, assuming that all of them are going to be sicker than the crew. It can be concluded that this model for Motion Sickness gives a reasonable measure to determine the amount of motion sickness personnel onboard the HSV-2 experienced during the period of data collection.

2. Operations Manual (OPSMAN)

With the validated model, the next step was to obtain a first approach to an operations manual that would be able to predict the amount of sickness that troops may experience when embarked on a vessel that has ride characteristics like the SWIFT. In order to generate this OPSMAN, a wide range of variables were taken in account in order to aid in the planning process of an operation. This is the first approach to an OPSMAN and in no way does this study predict

the performance of the crew or embarked troops after they have become motion sick. In other words, this OPSMAN will provide a guide to determine how motion sick personnel may be after certain exposures, not how effective they will be in their tasks after becoming motion sick. The performance of personnel after becoming motion sick is outside of the scope of this study.

The motion sickness model suggests that motion sickness is a function of wave height and exposure time. In general, the higher the wave height, the more motion sick personnel will be. Additionally, motion sickness incidence should decrease as exposure time increases and subjects become more adapted to the environment.

The MSAQ scores were broken down into quartiles representing four categories of motion sickness: Minimum, Moderate, Severe, and Extreme based. A table was developed with the upper and lower bounds for motion sickness score for sea states between 0 and 15 feet and for exposure times between 0 and 48 hrs. This gives the OPSMAN a lookup capability to determine expected lower bound levels of motion sickness for embarked personnel (see APPENDIX C). For example, if an operation takes the SWIFT or a vessel like it from port into seas with wave heights that are between 5 and 7 feet for a 15 hour transit, the expected lower bound MSAQ scores among the embarked personnel onboard is 16.20 with a maximum of 20.7 and a minimum of 11.7. This gives a Moderate Sick Score for the lower bound of the troops, in other words the best they will do is have a Moderate motion sickness. Figure 37 depicts a plot of the upper and lower bounds of the MSAQ score prediction model.

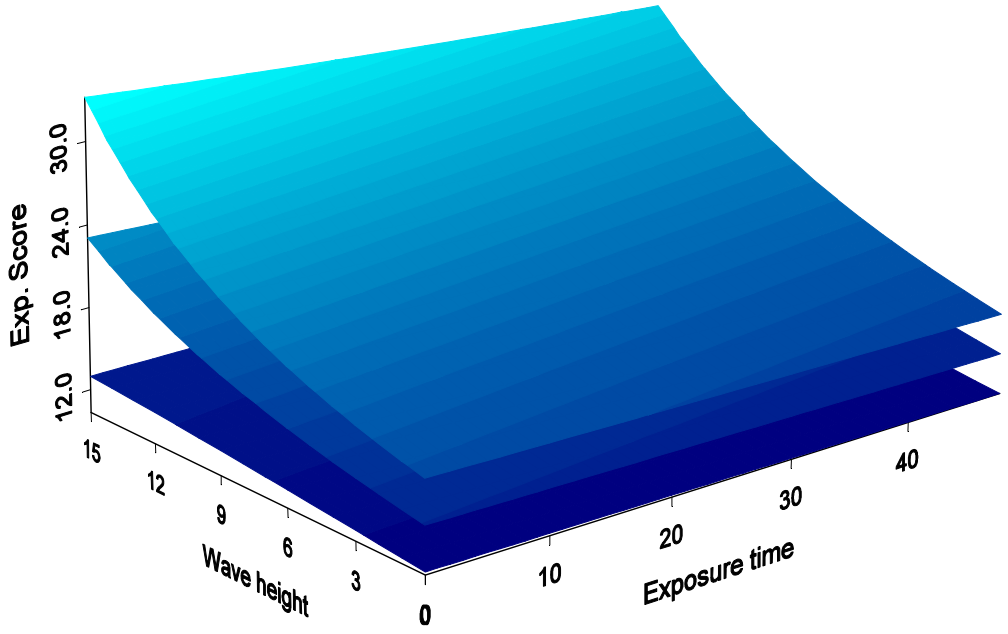

Figure 37. Operations Manual Graph

B. MOTION INDUCED TASK INTERRUPTION ANALYSIS

1. Simulation Output Analysis

Simulation output data was collected from each of the five simulation runs: the Engineering Rover run, the High Speed Open Ocean Transit run, the High Speed Littoral Transit run, the Anti Surface Warfare run, and the Naval Surface Fire Support run (see APPENDIX F). The watch stander utilization data from each repetition was organized in spreadsheet format along with the corresponding input parameters from the NOLH experiment design. Linear regression was used to analyze the data in order to determine the partial effects that interruption duration and frequency had on every watch stander in each of the ship missions.

For each model, variance transformations were used to stabilize the dependent variables prior to fitting the regressions. Multicolinearity between predictors was not a problem because of the nearly orthogonal arrays of values produced by the NOLH experiment design. Each regression fit began with the

complete linear regression model including all of the predictors for that mission. Predictors were removed one at a time from the complete model utilizing backwards elimination based on T test values while maintaining a 95% level of significance. Regression models for each watch stander are found in APPENDIX B.

a. Engineering Rover

Analysis of the ER output data yielded that the rover's utilization is effected by assigned tasks and not by MITI frequency or duration. The final linear model yielded an F-statistic value of 28.55 with a p-value of 0.00. R^2 for this model was 0.08 which means that the model is accurate, but does not explain the variability in the values for ER utilization. As the amount of time between interruptions decreases (frequency increases) and the length of the interruptions increases, the utilization of the ER only changes by very small amounts. This result goes to show that the ER utilization is driven by assigned tasks and not MITIs.

b. Navigator of the Watch

Analysis of the NOOW output data yielded that the utilization of the watch is dependant on both the frequency and length of interruptions in the presence of all other independent variables. The linear models that fit the NOOW for each mission area all included MITI frequency and length in them and had p-value of 0.00 with R^2 statistics of .80 or larger. A 95% confidence interval was produced by fixing the mean task occurrences for each of the mission areas in the simulation while varying the interruption lengths and frequencies (Figures 38, 39, 40, and 41). These confidence intervals display that the NOOW is busiest in the missions of ASUW and Littoral Transit (LT) followed by NSFS. Open ocean transit (OT) is the least busy mission which concurs with normal operations on a ship. All of the upper bounds for the confidence intervals fall below 50% which indicate that the Navigator of the Watch is probably able to

perform all assigned tasks under most work loads in spite of performance degradation due to motion induced task interruptions.

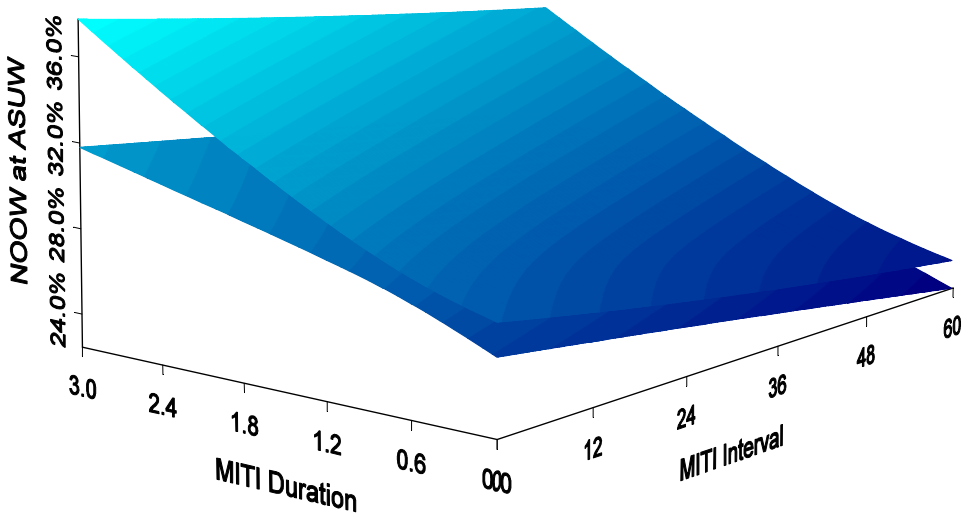

Figure 38. 95% CI for NOOW utilization at ASUW

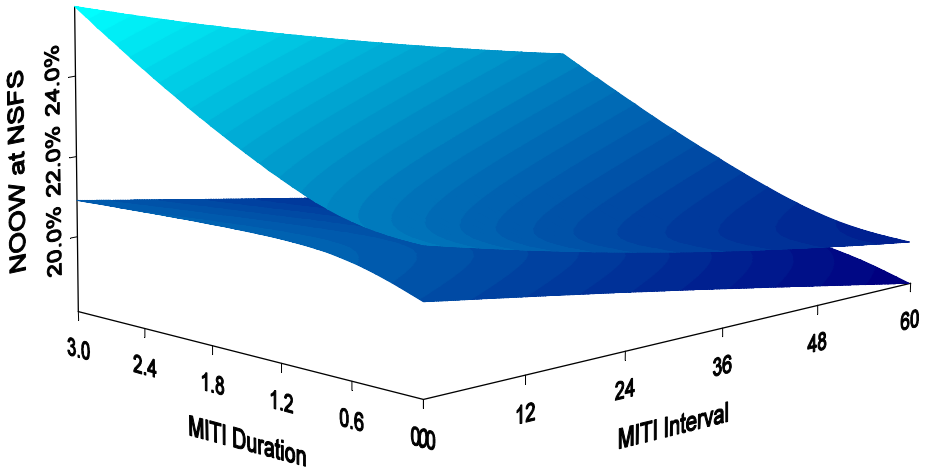

Figure 39. 95% CI for NOOW utilization at NSFS

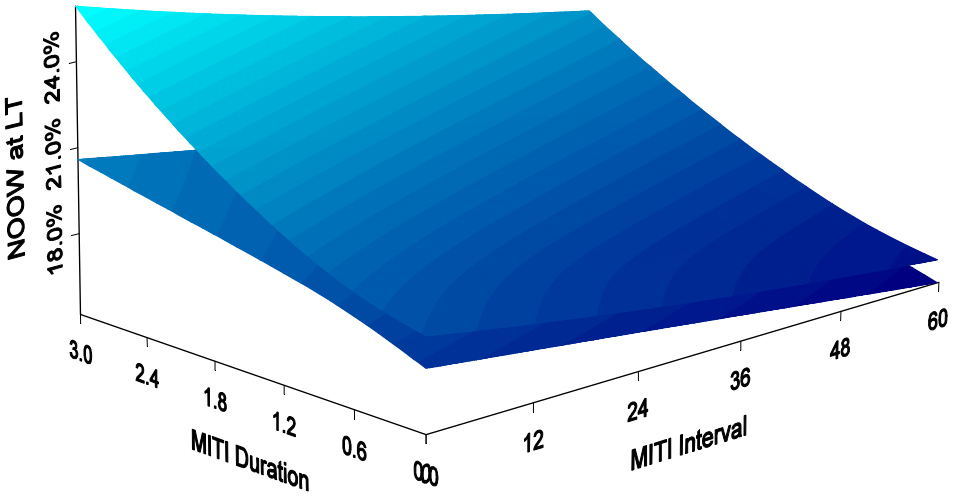

Figure 40. 95% CI for NOOW utilization at LT

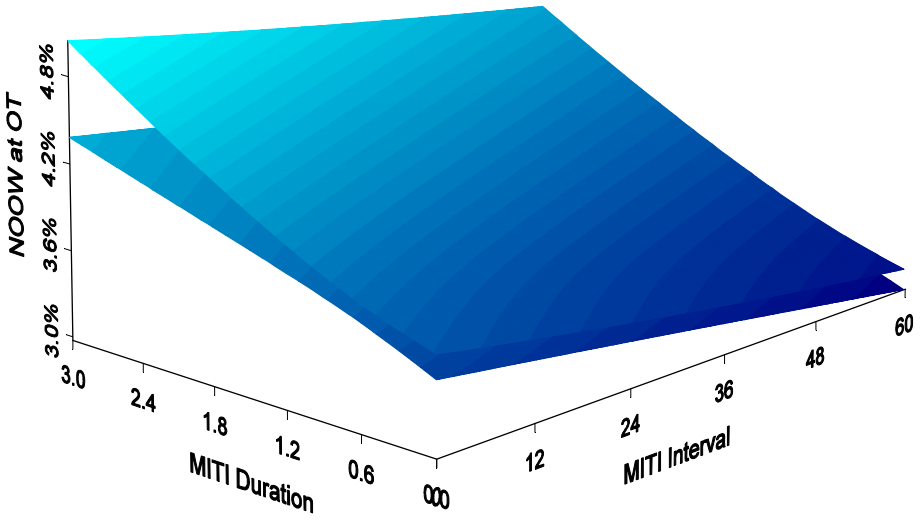

Figure 41. 95% CI for NOOW utilization at OT

c. Officer of the Deck

Analysis of the OOD output data found that the utilization of the watch is also dependant on both the frequency and length of interruptions in the presence of all other independent variables. The linear models that fit the OOD for each mission area included MITI frequency and length and had p-value of 0.00 with R^2 statistics of .80 or larger. A 95% confidence interval was produced

by fixing the mean task occurrences for each of the mission areas in the simulation while varying the interruption lengths and frequencies (Figures 42, 43, 44, and 45). These confidence intervals display that the OOD, like the NOOW, is busiest in the missions of ASUW and littoral transit. NSFS is the next busiest mission and open ocean transit is the least busy. As in the case of the NOOW, all of the upper bounds for the confidence intervals fall below 50% which indicate that the Officer of the Deck is probably able to perform all duties under most work loads in spite of performance degradation due to motion induced task interruptions.

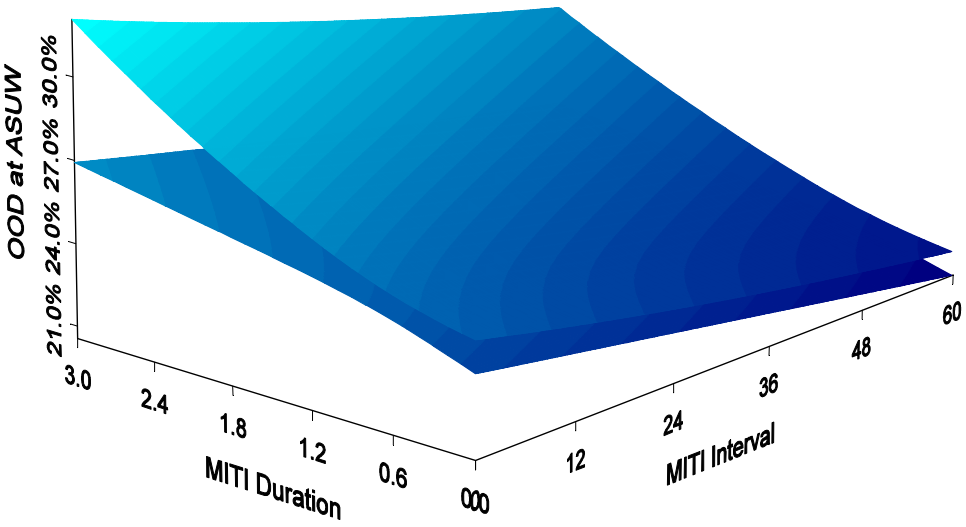

Figure 42. 95% CI for OOD utilization at ASUW

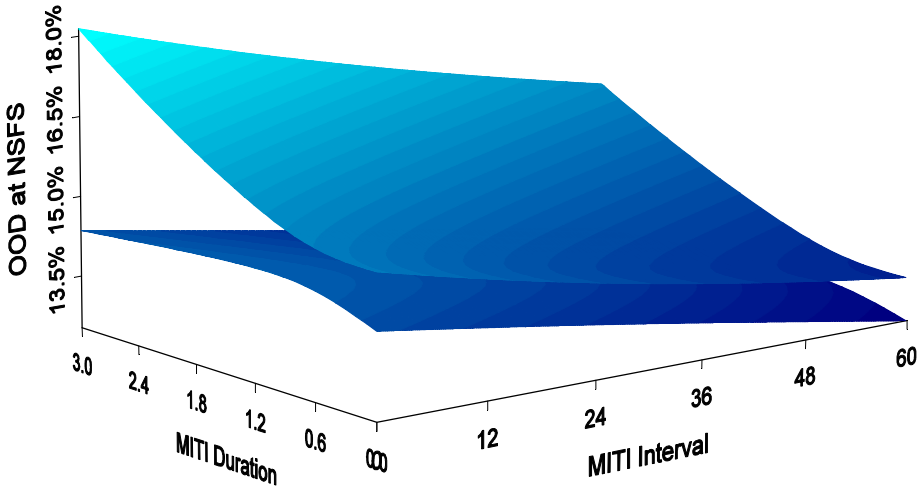

Figure 43. 95% CI for OOD utilization at NSFS

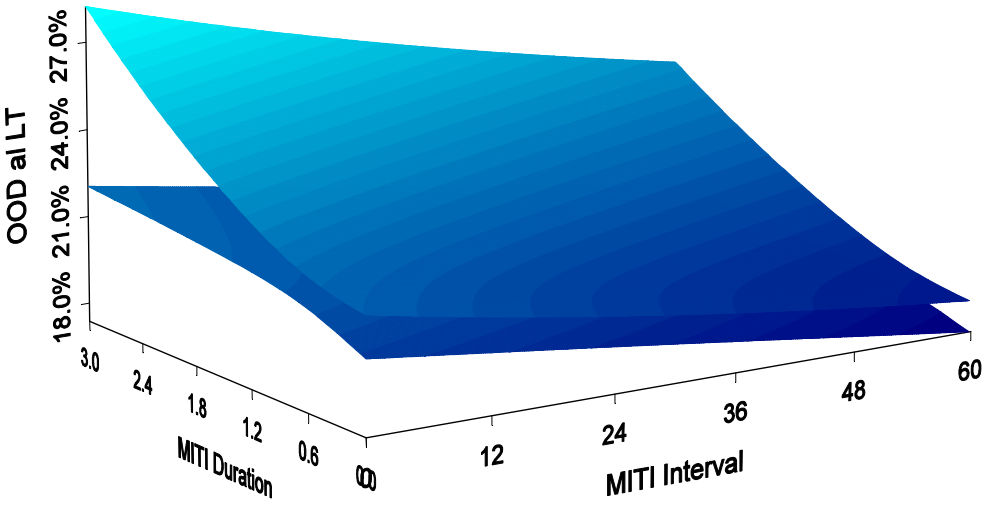

Figure 44. 95% CI for OOD utilization at LT

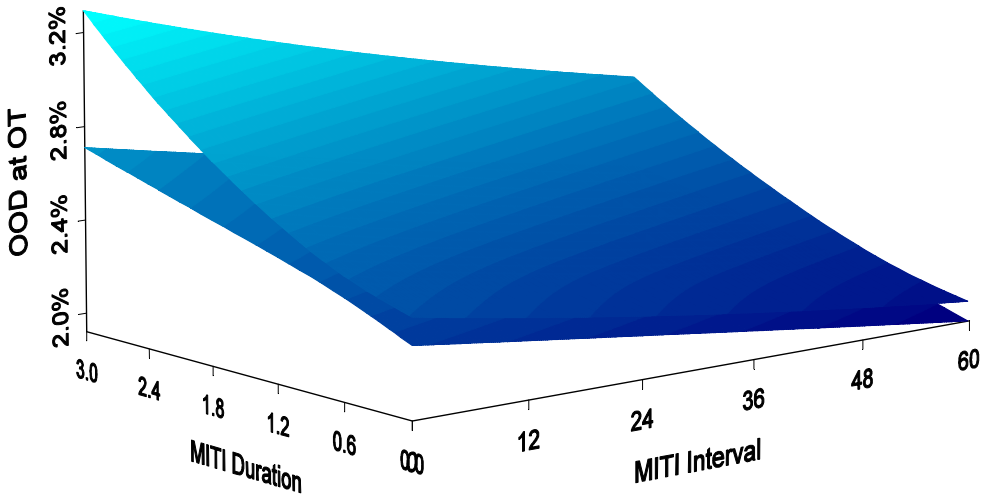

Figure 45. 95% CI for OOD utilization at OT

d. Tactical Action Officer

Analysis of the TAO output data yielded that the utilization of the watch is dependant on both the frequency and length of task interruptions in the presence of all other independent variables. The linear models that fit the TAO utilization for each mission area included MITI frequency and length and had p-value of 0.00 with R^2 statistics of 0.80 or larger. A 95% confidence interval was produced by fixing the mean task occurrences for each of the mission areas in the simulation while varying the interruption lengths and frequencies (Figure 46, 47, 48, and 49). These confidence intervals display that the TAO is busiest in the mission areas of ASUW and NSFS, which is intuitive since the TAO main responsibilities are tactical in nature. Utilization values for open ocean and littoral transit are both low for the TAO. As in the case of the other modeled watch standers, all of the upper bounds for the confidence intervals fall below 50% which indicate that the Tactical Action Officer is probably able to perform all duties under most work loads in spite of performance degradation due to motion induced task interruptions.

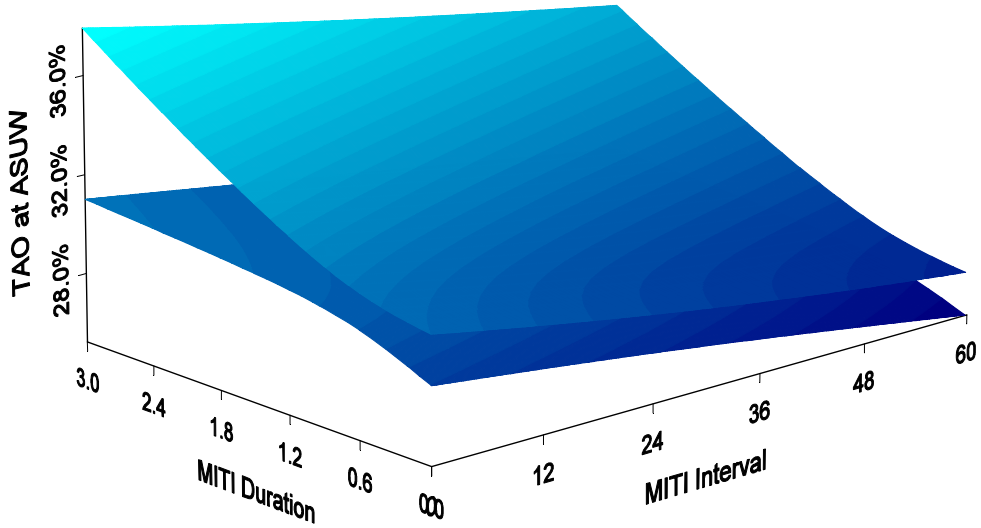

Figure 46. 95% CI for TAO utilization at ASUW

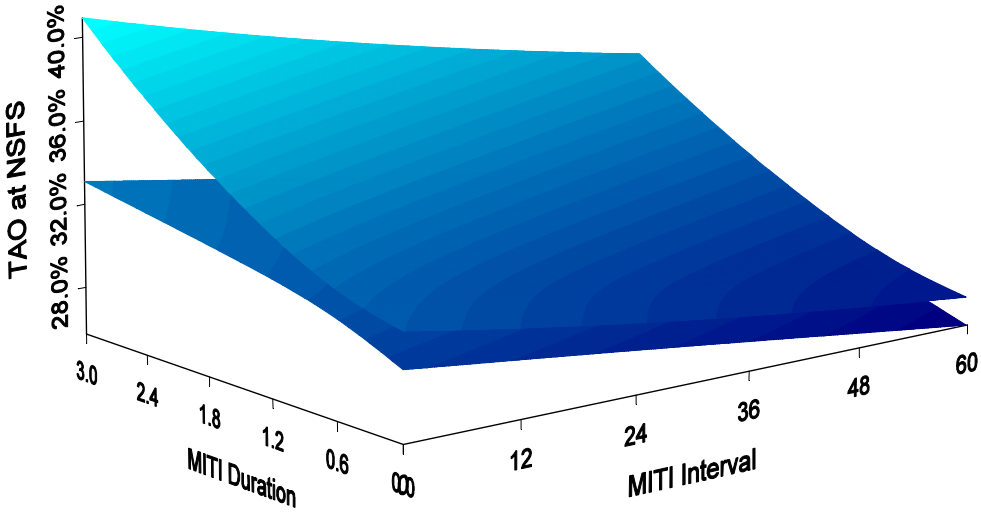

Figure 47. 95% CI for TAO utilization at NSFS

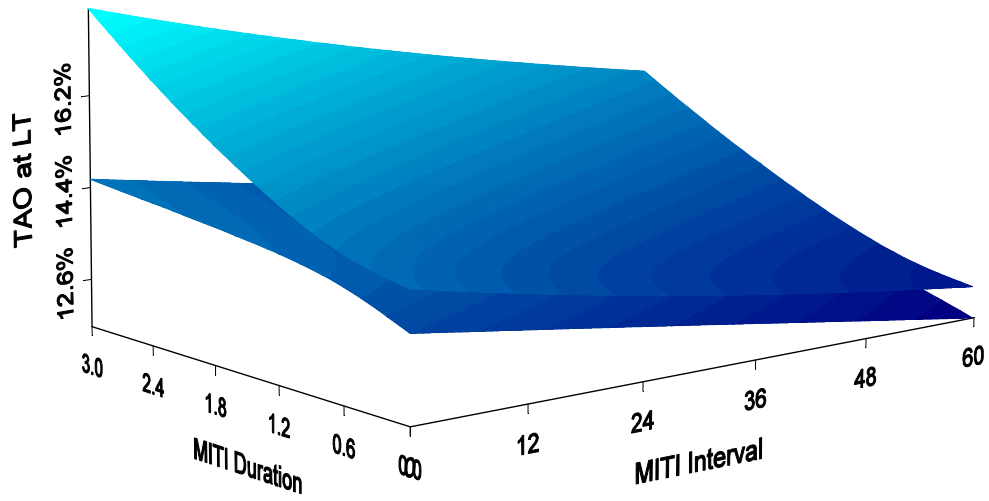

Figure 48. 95% CI for TAO utilization at LT

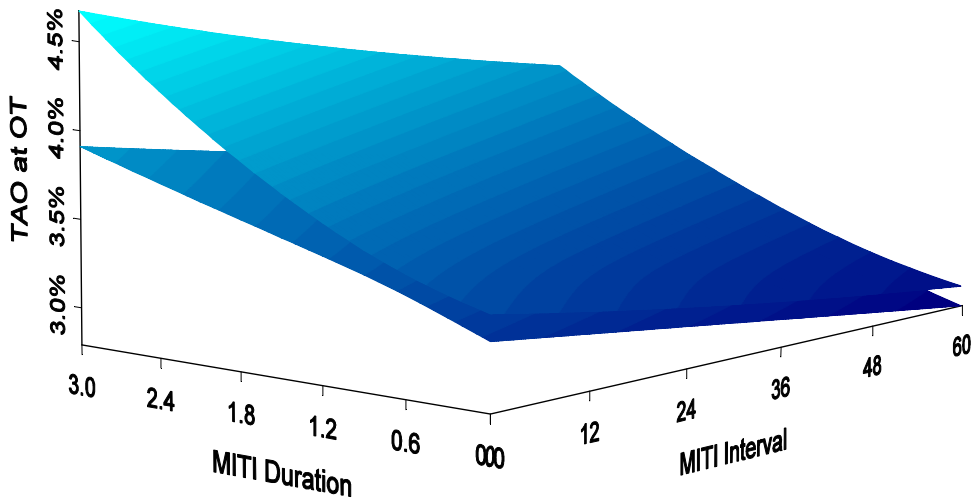

Figure 49. 95% CI for TAO utilization at OT

IV. CONCLUSIONS AND RECOMMENDATIONS

A. MOTION SICKNESS

The motion sickness model developed in this study is a good first approach to determining the amount of motion sickness that personnel embarked on a ship similar to SWIFT will experience. However, motion sickness is experienced in different ways by different individuals and not every person reacts the same way to the same stimuli. Because of the two elements that compose motion sickness, the physiological and the psychological, the ability for a particular individual to psychologically overcome motion sickness symptoms is not well known. For example, in the case of landing force personnel, the adrenaline of engaging in a combat situation upon debarkation may be sufficient to overcome or ignore the motion sickness symptoms. Personnel performance under these types of circumstances is still not well known and is not part of this study. It is difficult to accurately predict how motion sick a person will be after being exposed to vessel motion. But based on the data collected, the motion sickness model gives a good estimate of the motion sickness conditions that can be expected. The output of the Operations Manual is the best situation that can be expected for motion sickness in embarked personnel. Using the Operations Manual, a planning team may develop courses of action knowing that a landing force may experience a least these levels of motion sickness.

B. MOTION INDUCED TASK INTERRUPTIONS

The results from the HSV-2 simulation analysis provide insights into the way that MITIs affect watch standers' capabilities to complete their. In the case of the Engineering Rover, MITIs did not play a significant factor in accounting for watch utilization. In the cases of all other watch standers in every mission area,

increasing degrees of MITIs did account for decreased task performance. However, the degradations were not severe enough to raise utilization values to a level that may indicate that a watch stander may not be able to complete their duties.

While this simulation model has not been validated and should not be used for prediction, it does indicate that MITIs affect task performance. The 95% confidence intervals all had upper bounds below 50% for watch utilizations, but there were some isolated simulation results that had utilization values near 1.0 for the TAO and OOD in the mission areas of ASUW and NSFS. These occurred during periods of very frequent and long interruptions combined with high rates of occurrence of other events. Therefore, it is probable that all watches aboard SWIFT, or a future LCS class with similar ride characteristics, will be able to perform their duties in periods of heavy interruptions, but there is also a small possibility that they can become overwhelmed when the rates of occurrence of many required tasks become large. Mitigation for this possibility is to have additional watch standers to assist in routine tasks during complex missions such as ASUW and NSFS in order to allow primary watch standers to focus on the larger tasks at hand.

C. RECOMMENDATIONS FOR FURTHER STUDY

All data was collected while SWIFT was performing normal operations. In order to facilitate more complete data sets, a high speed vessel should be committed to testing alongside a control vessel. This would allow for collection of a full range of data in terms of wave heights, speeds, and other independent variables. It would also allow for collection efforts to include vertical acceleration frequency data that would enable the use of current motion sickness models such as the motion sickness dose model (Griffin, 1990) to facilitate analysis. It would also allow for comparison to data collected on the control ship to determine if personnel embarked on high speed vessels experience different

levels of motion sickness and degradations to task performance than those on traditional ships.

Another recommendation is to determine a more effective method for collecting motion sickness score data than paper surveys. Many of the issued surveys were missing data fields and some were not filled in at all. One cause of this is that when personnel experienced severe motion sickness, they stopped making entries in the MSAQ surveys. Less subject-dependent collection methods such as interviews, voice recorders, or personal data assistants programmed with data collection programs may provide more complete data sets in future studies.

A follow up study could be to attempt to gauge the effectiveness of landing force personnel given their state of motion sickness upon debarkation. This analysis could better explore the relationship between the physiological and psychological aspects of motion sickness.

Another follow up study could be an attempt to validate the MITI simulation model using dedicated watch standers and research personnel. More data also should be collected as to the frequencies of MITIs to determine an accurate model to better predict their rates of occurrence. Work can also be done in correlating the results of the MITI model with the previous MII models developed by Baitis, Graham, and Meyers (1991).

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. DATA SETS

A. CREW DATA APRIL 2005

Total Score	Subject	Exposure Time	Wave height	Ships Speed	Wave Direction	T-Foil
12.500	1	20	5	18.1	1	1
11.111	1	25	5	25.0	2	1
11.111	1	29	5	25.0	2	1
11.111	1	34	7	19.0	2	1
13.889	1	43	7	17.6	1	1
11.806	1	47	7	17.3	1	1
11.111	1	51	7	17.4	1	1
11.111	1	66	5	18.0	1	1
11.111	1	71	4	22.0	2	1
11.111	1	66	5	18.0	1	1
11.111	1	71	4	22.0	2	1
11.111	1	75	4	21.5	2	1
11.111	1	91	3	22.9	2	1
11.111	1	94	4	18.5	2	1
11.111	1	99	3	19.2	2	1
11.111	1	115	3	18.0	2	1
11.111	1	117	3	18.4	2	1
12.500	1	140	7	18.2	1	1
15.972	1	150	7	20.0	0	0
18.056	2	18	4	26.5	2	1
16.667	2	23	5	26.0	1	1
19.444	2	29	5	25.0	2	1
25.000	2	35	7	19.0	2	1
29.167	2	43	7	17.6	1	1
30.556	2	46	7	17.3	1	1
16.667	2	51	7	17.4	1	1
22.222	2	56	7	23.2	1	1
16.667	2	67	5	18.0	2	1
18.056	2	71	4	22.0	2	1
15.278	2	75	4	21.5	2	1
15.972	2	79	4	21.3	2	1
13.889	2	90	4	22.7	1	1
11.111	2	95	4	19.0	1	1
12.500	2	101	3	19.4	2	1
15.278	2	105	3	22.8	0	1
15.972	2	115	3	18.0	2	1
15.278	2	119	3	18.7	1	1
14.583	2	125	5	18.8	1	1

13.194	3	17	4	23.8	1	1
11.111	3	25	5	25.0	2	1
14.583	3	29	5	25.0	2	1
12.500	3	33	7	19.0	2	1
11.111	3	42	7	17.5	1	1
13.194	3	49	7	17.3	2	1
12.500	3	53	7	17.0	1	1
11.111	3	65	5	18.4	1	1
11.111	3	74	4	22.6	2	1
11.111	3	77	4	21.5	2	1
11.111	3	91	3	22.9	2	1
11.111	3	96	4	18.5	1	1
11.111	3	101	3	19.4	2	1
11.111	3	105	3	22.8	0	1
10.417	3	118	3	18.6	1	1
12.500	3	122	5	18.4	1	1
11.111	3	127	5	18.9	1	1
13.889	3	138	7	17.8	1	1
15.278	4	25	5	25.0	2	1
30.556	4	34	7	19.0	2	1
17.361	4	48	7	17.4	2	1
13.194	4	56	7	23.2	1	1
11.806	4	68	5	21.1	2	1
11.806	4	80	4	21.4	2	1
12.500	4	93	3	22.9	2	1
11.806	4	105	3	22.8	0	1
11.806	4	115	3	18.0	2	1
12.500	4	120	5	18.4	1	1
11.806	4	141	7	18.0	1	1
13.194	4	153	7	17.9	0	1
13.194	4	163	5	17.8	0	1
12.500	4	177	5	25.1	0	1
11.111	5	23	5	26.0	1	1
11.111	5	27	4	25.0	2	1
11.111	5	31	5	25.5	2	1
11.111	5	35	7	19.0	2	1
11.111	5	43	7	17.6	1	1
11.111	5	51	7	17.4	1	1
11.111	5	55	7	17.9	1	1
11.111	5	59	5	21.5	2	1
11.111	5	67	5	18.0	2	1
11.111	5	71	4	22.0	2	1
11.111	5	75	4	21.5	2	1
11.111	5	79	4	21.3	2	1
11.111	5	91	3	22.9	2	1
11.111	5	95	4	19.0	1	1
11.111	5	99	3	19.2	2	1
11.111	5	103	3	12.8	2	1
11.111	5	115	3	18.0	2	1
11.111	5	123	5	18.5	1	1

11.111	5	127	5	18.9	1	1
13.889	6	54	7	17.6	2	1
11.111	6	68	5	21.1	2	1
13.194	7	23	5	26.0	1	1
12.500	7	27	4	25.0	2	1
11.806	7	30	5	25.0	2	1
11.111	7	34	7	19.0	2	1
13.889	7	43	7	17.6	1	1
11.111	7	47	7	17.3	1	1
14.583	7	51	7	17.4	1	1
20.833	7	55	7	17.9	1	1
11.111	7	67	5	18.0	2	1
13.194	7	72	4	22.0	2	1
13.194	7	75	4	21.5	2	1
11.806	7	79	4	21.3	2	1
27.083	7	91	3	22.9	2	1
11.111	8	20	5	18.1	1	1
11.111	8	23	5	26.0	1	1
11.111	8	28	5	25.0	2	1
11.111	8	43	7	17.6	1	1
11.806	8	47	7	17.3	1	1
11.111	8	67	5	18.0	2	1
11.111	8	71	4	22.0	2	1
11.111	8	77	4	21.5	2	1
13.194	8	90	4	22.7	1	1
11.806	8	95	4	19.0	1	1
11.111	8	101	3	19.4	2	1
11.111	8	113	3	19.1	2	1
10.417	8	119	3	18.7	1	1
11.111	8	123	5	18.5	1	1
11.111	8	138	7	17.8	1	1
11.111	8	150	7	20.0	0	0
13.889	9	24	5	25.5	1	1
11.111	9	29	5	25.0	2	1
11.111	9	41	8	17.5	2	1
15.278	9	48	7	17.4	2	1
14.583	9	65	5	18.4	1	1
14.583	9	72	4	22.0	2	1
11.111	9	81	4	21.3	2	1
13.889	9	89	4	22.7	1	1
15.972	9	94	4	18.5	2	1
11.111	9	101	3	19.4	2	1
11.111	9	105	3	22.8	0	1
13.194	9	113	3	19.1	2	1
11.111	9	120	5	18.4	1	1
11.111	9	125	5	18.8	1	1
11.111	9	129	5	18.4	1	1
11.111	9	138	7	17.8	1	1
11.111	9	144	7	18.0	1	1
11.111	9	149	7	19.2	0	0

11.111	9	153	7	17.9	0	1
23.611	10	21	5	25.5	1	1
21.528	10	28	5	25.0	2	1
28.472	10	57	6	20.8	2	1
20.833	10	41	8	17.5	2	1
21.528	10	45	7	17.4	1	1
16.667	10	51	7	17.4	1	1
19.444	10	58	6	22.5	2	1
13.889	10	65	5	18.4	1	1
13.194	10	70	4	22.0	2	1
13.889	10	76	4	21.3	2	1
18.056	10	82	4	21.9	2	1
15.278	10	89	4	22.7	1	1
18.750	10	96	4	18.5	1	1
16.667	10	102	3	13.2	2	1
16.667	10	108	3	20.0	2	1
11.806	10	115	3	18.0	2	1
11.806	10	120	5	18.4	1	1
30.556	10	130	5	19.3	1	1
28.472	10	139	7	17.7	1	1
11.111	11	25	5	25.0	2	1
11.111	11	30	5	25.0	2	1
24.306	11	39	8	17.4	1	1
22.222	11	44	7	17.8	1	1
11.111	11	53	7	17.0	1	1
11.111	11	67	5	18.0	2	1
11.111	11	73	4	21.6	2	1
11.111	11	78	4	21.6	2	1
11.111	11	84	4	21.8	1	1
11.111	11	91	3	22.9	2	1
11.111	11	97	4	18.6	1	1
11.111	11	105	3	22.8	0	1
11.111	11	115	3	18.0	2	1
11.111	11	119	3	18.7	1	1
11.111	11	127	5	18.9	1	1
13.194	11	139	7	17.7	1	1
12.500	11	142	7	18.5	1	1
11.111	11	164	5	17.3	0	1
11.111	11	170	0	0.0	0	0
11.111	12	23	5	26.0	1	1
11.111	12	30	5	25.0	2	1
11.111	12	33	7	19.0	2	1
11.111	12	42	7	17.5	1	1
11.111	12	47	7	17.3	1	1
11.111	12	50	7	17.5	2	1
11.111	12	57	6	20.8	2	1
11.111	12	65	5	18.4	1	1
11.111	12	71	4	22.0	2	1
11.111	12	75	4	21.5	2	1
11.111	12	81	4	21.3	2	1

11.111	12	90	4	22.7	1	1
11.111	12	94	4	18.5	2	1
11.111	12	100	3	19.3	2	1
11.111	12	106	3	22.5	0	1
11.111	12	116	3	18.8	2	1
11.111	12	119	3	18.7	1	1
11.111	12	124	5	18.9	1	1
15.278	13	28	5	25.0	2	1
16.667	13	42	7	17.5	1	1
13.889	13	50	7	17.5	2	1
11.806	13	66	5	18.0	1	1
11.111	13	71	4	22.0	2	1
11.111	13	77	4	21.5	2	1
11.111	13	90	4	22.7	1	1
12.500	13	95	4	19.0	1	1
11.111	13	99	3	19.2	2	1
11.111	13	115	3	18.0	2	1
11.111	13	123	5	18.5	1	1
11.111	13	129	5	18.4	1	1
11.111	13	140	7	18.2	1	1
11.111	13	143	7	18.5	1	1
11.111	13	147	7	18.0	0	1
11.111	13	162	5	17.6	0	1
11.111	13	167	5	7.8	0	1
11.111	13	173	0	0.0	0	0
11.111	13	186	3	26.3	0	0
11.111	14	27	4	25.0	2	1
11.111	14	31	5	25.5	2	1
11.111	14	42	7	17.5	1	1
11.111	14	47	7	17.3	1	1
11.111	14	52	7	18.0	1	1
11.111	14	57	6	20.8	2	1
11.111	14	66	5	18.0	1	1
11.111	14	72	4	22.0	2	1
11.111	14	76	4	21.3	2	1
11.111	14	81	4	21.3	2	1
11.111	14	89	4	22.7	1	1
11.111	14	95	4	19.0	1	1
11.111	14	99	3	19.2	2	1
11.111	14	105	3	22.8	0	1
11.111	14	113	3	19.1	2	1
11.111	14	119	3	18.7	1	1
11.111	14	123	5	18.5	1	1
11.111	14	129	5	18.4	1	1
11.111	14	138	7	17.8	1	1
13.194	15	21	5	25.5	1	1
11.111	15	27	4	25.0	2	1
11.111	15	30	5	25.0	2	1
13.194	15	34	7	19.0	2	1
11.111	15	42	7	17.5	1	1

11.111	15	45	7	17.4	1	1
11.111	15	50	7	17.5	2	1
12.500	15	53	7	17.0	1	1
11.111	15	57	6	20.8	2	1
11.111	15	66	5	18.0	1	1
11.111	15	70	4	22.0	2	1
11.111	15	74	4	22.6	2	1
11.111	15	77	4	21.5	2	1
13.194	15	81	4	21.3	2	1
11.111	15	89	4	22.7	1	1
11.111	15	93	3	22.9	2	1
11.111	15	100	3	19.3	2	1
12.500	15	105	3	22.8	0	1
11.111	15	114	3	18.9	2	1
11.111	16	20	5	18.1	1	1
11.111	16	23	5	26.0	1	1
11.111	16	27	4	25.0	2	1
11.111	16	40	8	17.0	1	1
11.111	16	42	7	17.5	1	1
11.111	16	47	7	17.3	1	1
11.111	16	56	7	23.2	1	1
11.111	16	67	5	18.0	2	1
11.111	16	90	4	22.7	1	1
11.111	16	93	3	22.9	2	1
11.111	16	94	4	18.5	2	1
11.111	16	102	3	13.2	2	1
11.111	16	106	3	22.5	0	1
11.111	16	114	3	18.9	2	1
11.111	16	118	3	18.6	1	1
11.111	16	123	5	18.5	1	1
11.111	16	138	7	17.8	1	1
11.111	16	144	7	18.0	1	1
11.111	16	162	5	17.6	0	1
12.500	17	34	7	19.0	2	1
15.972	17	39	8	17.4	1	1
13.194	17	71	4	22.0	2	1
11.806	17	95	4	19.0	1	1
11.806	17	119	3	18.7	1	1
12.500	17	143	7	18.5	1	1
11.111	17	167	5	7.8	0	1

B. MARINES DATA APRIL 2005

Total Score	Subject	Exposure Time	Wave height	Ships Speed	Wave Direction	T-Foil
47.917	1	1	9	10.0	1	1
68.750	1	2	9	17.0	1	1
61.111	1	4	10	17.0	1	1

34.028	2	2	9	17.0	1	1
37.500	2	3	9	17.0	1	1
22.222	2	4	10	17.0	1	1
29.167	2	5	10	15.0	1	1
40.278	2	6	10	15.0	1	1
40.972	2	7	10	17.0	1	1
34.722	2	8	10	17.0	1	1
70.139	3	1	9	10.0	1	1
70.833	3	4	10	17.0	1	1
58.333	3	7	10	17.0	1	1
25.000	4	1	9	10.0	1	1
18.750	4	2	9	17.0	1	1
20.139	4	4	10	17.0	1	1
3.472	4	7	10	17.0	1	1
5.556	4	8	10	17.0	1	1
27.083	5	1	9	10.0	1	1
27.083	5	2	9	17.0	1	1
33.333	5	3	9	17.0	1	1
22.222	5	5	10	15.0	1	1
17.361	5	6	10	15.0	1	1
18.750	5	7	10	17.0	1	1
18.056	5	10	10	8.0	0	1
18.750	6	1	9	10.0	1	1
19.444	6	2	9	17.0	1	1
20.833	6	3	9	17.0	1	1
24.306	6	4	10	17.0	1	1
19.444	6	5	10	15.0	1	1
18.750	6	6	10	15.0	1	1
18.750	6	7	10	17.0	1	1
14.583	6	8	10	17.0	1	1
25.694	7	1	9	10.0	1	1
20.833	7	3	9	17.0	1	1
22.222	7	6	10	15.0	1	1
22.917	7	8	10	17.0	1	1
11.111	7	10	10	8.0	0	1
21.528	8	1	9	10.0	1	1
31.250	8	2	9	17.0	1	1
20.139	8	3	9	17.0	1	1
26.389	8	4	10	17.0	1	1
18.056	8	5	10	15.0	1	1
18.056	8	7	10	17.0	1	1
13.194	8	8	10	17.0	1	1
20.833	9	0	9	10.0	1	1
13.889	9	1	9	10.0	1	1
18.750	9	2	9	17.0	1	1
20.833	9	3	9	17.0	1	1
16.667	9	4	10	17.0	1	1
20.833	9	5	10	15.0	1	1
20.833	9	6	10	15.0	1	1
15.972	9	7	10	17.0	1	1

25.694	9	10	10	8.0	0	1
13.889	10	1	9	10.0	1	1
15.972	10	2	9	17.0	1	1
18.750	10	5	10	15.0	1	1
18.750	10	7	10	17.0	1	1
27.778	10	10	10	8.0	0	1
12.500	11	1	9	10.0	1	1
16.667	11	2	9	17.0	1	1
15.278	11	7	10	17.0	1	1
11.111	11	10	10	8.0	0	1
18.056	12	1	9	10.0	1	1
22.222	12	2	9	17.0	1	1
16.667	12	3	9	17.0	1	1
11.806	12	4	10	17.0	1	1
11.111	12	5	10	15.0	1	1
11.111	12	6	10	15.0	1	1
16.667	12	7	10	17.0	1	1
16.667	12	8	10	17.0	1	1
100.000	13	1	9	10.0	1	1
88.889	13	2	9	17.0	1	1
73.611	13	7	10	17.0	1	1
55.556	13	8	10	17.0	1	1
15.278	14	1	9	10.0	1	1
21.528	14	2	9	17.0	1	1
18.056	14	7	10	17.0	1	1
20.833	14	10	10	8.0	0	1
29.861	15	1	9	10.0	1	1
39.583	15	2	9	17.0	1	1
22.222	15	7	10	17.0	1	1
12.500	16	1	9	10.0	1	1
12.500	16	2	9	17.0	1	1
14.583	16	7	10	17.0	1	1
13.194	16	9	10	17.0	1	1
13.194	17	1	9	10.0	1	1
32.639	17	2	9	17.0	1	1
29.167	17	3	9	17.0	1	1
18.750	17	4	10	17.0	1	1
15.278	17	5	10	15.0	1	1
13.889	17	6	10	15.0	1	1
18.750	17	8	10	17.0	1	1
11.806	18	1	9	10.0	1	1
11.111	18	2	9	17.0	1	1
11.111	18	3	9	17.0	1	1
11.111	18	4	10	17.0	1	1
11.111	18	5	10	15.0	1	1
11.111	18	6	10	15.0	1	1
11.111	18	7	10	17.0	1	1
11.111	18	10	10	8.0	0	1
29.167	19	0	9	10.0	1	1
23.611	19	1	9	10.0	1	1

27.778	19	3	9	17.0	1	1
29.861	19	5	10	15.0	1	1
15.278	20	1	9	10.0	1	1
18.056	20	2	9	17.0	1	1
11.111	20	3	9	17.0	1	1
11.111	20	4	10	17.0	1	1
11.111	20	5	10	15.0	1	1
13.194	20	6	10	15.0	1	1
15.278	20	7	10	17.0	1	1
15.972	20	8	10	17.0	1	1
18.056	20	9	10	17.0	1	1
73.611	21	1	9	10.0	1	1
75.000	21	3	9	17.0	1	1
76.389	21	5	10	15.0	1	1
76.389	21	6	10	15.0	1	1
72.222	21	7	10	17.0	1	1

C. MITI LENGTH DATA

MITI length	Z acel	Wave high	Speed	Wave Dir	t-Foil	Position	Braced	Camera
8	0.219062	8	17.5	2	1	2	0	2
6	0.201933	8	17.5	2	1	2	0	2
2	0.201933	8	17.5	2	1	2	0	2
13	0.472757	7	17.5	1	1	3	1	1
60	0.472757	7	17.5	1	1	3	1	1
71	0.472757	7	17.5	1	1	3	1	1
7	0.472757	7	17.5	1	1	2	1	2
7	0.472757	7	17.5	1	1	2	0	2
14	0.303202	7	17.5	1	1	1	0	1
14	0.303202	7	17.5	1	1	1	0	1
13	0.174847	7	17.5	1	1	1	0	1
8	0.174847	7	17.5	1	1	1	0	1
10	0.193515	7	17.5	1	1	2	0	1
4	0.141949	7	17.5	1	1	2	0	2
34	0.258615	7	17.5	1	1	2	0	1
8	0.191854	7	17.5	1	1	2	0	1
3	0.176694	7	17.5	1	1	2	0	2
10	0.215121	7	17.5	1	1	2	0	2
13	0.230537	7	17.6	1	1	1	1	1
13	0.230537	7	17.6	1	1	4	1	1
39	0.552574	7	17.6	1	1	3	1	1
4	0.200906	7	17.8	1	1	1	0	0
13	0.160964	7	17.8	1	1	1	0	0
13	0.160964	7	17.8	1	1	1	0	0
13	0.160964	7	17.8	1	1	1	0	0

10	0.203605	7	17.4	1	1	1	0	1
13	0.171312	7	17.4	1	1	2	0	0
15	0.178268	7	17.4	1	1	1	0	1
15	0.178268	7	17.4	1	1	1	0	1
4	0.155971	7	17.3	1	1	2	0	2
4	0.212089	7	17.3	1	1	1	0	2
3	0.158848	7	17.3	1	1	2	0	2
3	0.158848	7	17.3	1	1	2	1	2
55	0.201416	7	17.4	2	1	3	1	1
62	0.238977	7	17.4	2	1	2	1	2
9	0.171723	7	17.4	2	1	1	1	2
4	0.203317	7	17.4	1	1	2	0	1
36	0.200912	7	17.4	1	1	1	0	2
35	0.232414	7	17.4	1	1	1	0	2
4	0.187345	7	17.4	1	1	2	0	1
11	0.187345	7	17.4	1	1	2	0	1
21	0.16219	7	18	1	1	3	1	1
21	0.16219	7	18	1	1	3	1	1
21	0.16219	7	18	1	1	2	0	1
21	0.16219	7	18	1	1	3	1	1
4	0.194248	7	18	1	1	2	0	1
61	0.320239	7	18	1	1	3	1	1
6	0.320239	7	18	1	1	2	0	1
114	0.675026	7	18	1	1	3	1	1
32	0.675026	7	18	1	1	3	1	1
4	0.194735	7	18	1	1	3	1	1
6	0.185535	7	18	1	1	1	0	1
6	0.185535	7	18	1	1	3	1	1
6	0.185535	7	18	1	1	3	1	1
5	0.194746	7	18	1	1	1	1	1
4	0.228341	7	18	1	1	3	1	1
4	0.205323	7	18	1	1	1	0	1
9	0.17102	7	18	1	1	1	0	1
9	0.17102	7	18	1	1	1	0	1
5	0.154649	7	17	1	1	1	0	1
4	0.182216	7	17	1	1	1	0	1
5	0.184577	7	17	1	1	1	0	1
4	0.174479	7	17	1	1	1	0	1
3	0.197884	7	17	1	1	1	1	2
5	0.193481	7	18.2	1	1	2	1	3
7	0.235034	7	18.2	1	1	2	1	3
8	0.235034	7	18.2	1	1	2	0	3
4	0.35337	7	18	1	1	2	0	3
12	0.268408	7	18	1	1	2	0	3
9	0.301342	7	18	1	1	2	0	3
7	0.209265	7	18	1	1	2	0	3
5	0.24829	7	18	1	1	1	0	3
7	0.245762	7	18	1	1	2	0	3
9	0.260912	7	18.5	1	1	2	0	3
6	0.260912	7	18.5	1	1	4	0	3

10	0.229721	7	18.5	1	1	2	0	3
5	0.253867	7	18.5	1	1	2	0	3
4	0.253867	7	18.5	1	1	2	0	3
4	0.224384	7	18.5	1	1	2	0	3
16	0.224089	7	17.7	1	1	2	0	2
6	0.237333	7	17.7	1	1	2	0	0
4	0.294229	7	17.7	1	1	2	0	0
5	0.170396	7	17.8	1	1	2	0	0
8	0.260289	7	17.8	1	1	2	0	0
2	0.187064	7	17.8	1	1	2	0	2
16	0.23637	7	17.8	1	1	1	1	0
3	0.25765	7	17.8	1	1	2	0	2
1	0.298039	7	18	0	1	2	1	0
3	0.262165	7	18	0	1	2	0	0
3	0.286427	7	18	0	1	2	0	0
3	0.254093	7	18	0	1	1	0	0
5	0.281064	7	18.5	0	1	1	1	0
6	0.226385	7	18.5	0	1	1	0	0
3	0.266844	7	19.2	0	0	2	0	2
2	0.453485	7	19.2	0	0	2	0	2
7	0.470631	7	19.2	0	0	2	0	2
25	0.447268	7	19.2	0	0	1	0	2
4	0.337637	7	19.2	0	0	1	0	0
4	0.337637	7	19.2	0	0	3	0	0
3	0.339594	7	19.2	0	0	2	1	0
13	0.325732	7	20	0	0	2	0	1
4	0.328573	7	20	0	0	2	0	1
5	0.449733	7	20	0	0	1	1	0
9	0.50263	7	20	0	0	3	1	0
28	0.931295	7	20	0	0	3	1	0
28	0.931295	7	20	0	0	3	1	0
28	0.931295	7	20	0	0	3	1	0
28	0.931295	7	20	0	0	2	1	0
28	0.931295	7	20	0	0	2	1	0
102	0.760595	7	20	0	0	1	1	2
7	0.320764	7	20	0	0	1	0	0
3	0.354536	7	20	0	0	2	0	1
10	0.243213	7	20	0	0	2	0	1
2	0.213388	7	18.1	0	0	2	0	0
2	0.447102	7	18.1	0	0	2	0	0
7	0.232801	7	18.1	0	0	1	0	0
32	0.241534	7	18.1	0	0	1	0	0
32	0.241534	7	18.1	0	0	1	0	0
31	0.267355	7	18	0	1	1	1	0
8	0.298218	7	18	0	1	1	0	0
6	0.265016	7	18	0	1	2	0	0

D. MITI FREQUENCY DATA

Freq	Wave high	Speed	Wave Dir	t-Foil
70	7	19.0	2	1
14	5	21.5	2	1
3	8	17.0	1	1
1	8	17.0	1	1
3	8	17.0	1	1
2	8	17.4	1	1
8	8	17.0	1	1
3	8	17.5	2	1
3	7	17.5	1	1
13	7	17.6	1	1
4	7	17.8	1	1
1	7	17.4	1	1
6	7	17.3	1	1
2	7	17.3	1	1
1	7	17.4	2	1
0	7	17.3	2	1
0	7	17.5	2	1
15	7	17.4	1	1
1	7	18.0	1	1
0	7	17.0	1	1
6	7	17.6	2	1
0	7	17.9	1	1
0	7	23.2	1	1
0	6	20.8	2	1
0	6	22.5	2	1
5	5	18.2	1	1
7	6	17.6	1	1
25	7	17.8	1	1
36	7	17.7	1	1
21	7	18.1	1	1
31	7	17.8	1	1
18	7	17.7	1	1
18	7	18.2	1	1
24	7	18.0	1	1
23	7	18.5	1	1
11	7	18.5	1	1
13	7	18.0	1	1
29	6	17.7	1	1
27	7	17.8	1	1
42	7	18.0	0	1
148	7	18.5	0	1
122	7	19.2	0	0
66	7	20.0	0	0
69	7	18.1	0	0
88	7	18.0	0	1
117	7	17.9	0	1

137	7	11.8	0	1
88	7	11.1	0	1
85	5	18.0	1	0
51	5	18.2	1	1

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. STATISTICAL MODELS

A. MOTION SICKNESS MODELS

1. Multicolinearity

```
      *** Linear Model for Exposure Time***

Call: lm(formula = Exposure.Time ~ Wave.height + Ships.Speed, data =
Regression.Data.Crew, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q   Max
 -69.98 -24.52 -4.215 13.43 124.4

Coefficients:
                Value Std. Error   t value  Pr(>|t|)
(Intercept)  233.6515   14.4040    16.2213    0.0000
Wave.height   -9.3951    1.2990    -7.2325    0.0000
Ships.Speed   -5.3421    0.5975    -8.9403    0.0000

Residual standard error: 33.28 on 275 degrees of freedom
Multiple R-Squared: 0.2975
F-statistic: 58.24 on 2 and 275 degrees of freedom, the p-value is 0
VIF = 1.42348



      *** Linear Model for Wave High***

Call: lm(formula = Wave.high ~ Exposure.Time + Ships.Speed, data =
Regression.Data.Crew, na.action = na.exclude)
Residuals:
    Min      1Q  Median  3Q   Max
 -6.291 -0.8731 -0.3674 1.1 3.262

Coefficients:
                Value Std. Error  t value Pr(>|t|)
  (Intercept)   9.1832   0.6545    14.0302   0.0000
Exposure.Time  -0.0170   0.0024    -7.2325   0.0000
  Ships.Speed  -0.1447   0.0275    -5.2554   0.0000

Residual standard error: 1.416 on 275 degrees of freedom
Multiple R-Squared: 0.1761
F-statistic: 29.39 on 2 and 275 degrees of freedom, the p-value is 2.7e-012
VIF = 1.213739



      *** Linear Model for Ship Speed***

Call: lm(formula = Ships.Speed ~ Exposure.Time + Wave.height, data =
Regression.Data.Crew, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q  Max
 -19.26 -1.683 0.5687 2.205 9.61

Coefficients:
```

```
                Value Std. Error  t value Pr(>|t|)
  (Intercept)  26.4228   0.8154    32.4029   0.0000
Exposure.Time  -0.0422   0.0047    -8.9403   0.0000
  Wave.height  -0.6307   0.1200    -5.2554   0.0000

Residual standard error: 2.956 on 275 degrees of freedom
Multiple R-Squared: 0.2402
F-statistic: 43.48 on 2 and 275 degrees of freedom, the p-value is 0
VIF = 1.316135
```

2. Variance

a. 1st Linear Model for MOSIC

```
             *** Linear Model ***

Call: lm(formula = Total.Score ~ (Exposure.Time + Wave.High + Ships.Speed +
Wave.Direction)^2, data = Mosick.Data.Crew, na.action =
      na.exclude)
Residuals:
    Min     1Q Median     3Q   Max
 -4.758 -1.435 -1.039 0.2885 17.79

Coefficients:
                              Value Std. Error  t value Pr(>|t|)
                (Intercept)  -0.4160  12.7100    -0.0327   0.9739
               Exposure.Time   0.0723   0.0954     0.7571   0.4497
                   Wave.High   1.7940   1.3193     1.3598   0.1751
                 Ships.Speed   0.1974   0.4939     0.3996   0.6898
            Wave.Direction1  -2.8027   6.8895    -0.4068   0.6845
            Wave.Direction2  -0.1223   3.2680    -0.0374   0.9702
     Exposure.Time:Wave.High  -0.0126   0.0069    -1.8309   0.0683
   Exposure.Time:Ships.Speed  -0.0006   0.0037    -0.1646   0.8694
Exposure.TimeWave.Direction1   0.0154   0.0212     0.7274   0.4676
Exposure.TimeWave.Direction2   0.0062   0.0132     0.4725   0.6369
       Wave.High:Ships.Speed   0.0080   0.0538     0.1491   0.8816
    Wave.HighWave.Direction1  -0.0931   0.3332    -0.2794   0.7801
    Wave.HighWave.Direction2  -0.0783   0.2798    -0.2799   0.7798
  Ships.SpeedWave.Direction1   0.0726   0.2539     0.2860   0.7751
  Ships.SpeedWave.Direction2  -0.0041   0.0822    -0.0502   0.9600

Residual standard error: 3.594 on 263 degrees of freedom
Multiple R-Squared: 0.0914
F-statistic: 1.89 on 14 and 263 degrees of freedom, the p-value is 0.02764
```

b. Sick Score > 11.5 Model for MOSIC

```
      *** Linear Model ***

Call: lm(formula = Total.Score ~ (Exposure.Time + Wave.High + Ships.Speed +
Wave.Direction)^2, data = Mosick.Data.Crew, subset = Total.Score >
      11.5, na.action = na.exclude)
Residuals:
    Min     1Q Median   3Q   Max
 -6.332 -2.765 -1.198 1.36 14.98

Coefficients:
```

```
                                Value Std. Error  t value Pr(>|t|)
                  (Intercept) -10.8745  38.5722    -0.2819   0.7787
                Exposure.Time   0.1771   0.2264     0.7823   0.4361
                    Wave.High   3.1853   4.7428     0.6716   0.5036
                  Ships.Speed   0.4129   1.5775     0.2617   0.7941
              Wave.Direction1  -1.2499  18.0049    -0.0694   0.9448
              Wave.Direction2  -1.4103   8.4430    -0.1670   0.8677
     Exposure.Time:Wave.High  -0.0265   0.0150    -1.7677   0.0805
   Exposure.Time:Ships.Speed  -0.0024   0.0082    -0.2952   0.7685
Exposure.TimeWave.Direction1   0.0373   0.0541     0.6899   0.4920
Exposure.TimeWave.Direction2   0.0141   0.0311     0.4535   0.6513
       Wave.High:Ships.Speed   0.0291   0.2056     0.1415   0.8878
    Wave.HighWave.Direction1  -0.5405   1.1615    -0.4653   0.6428
    Wave.HighWave.Direction2  -0.2806   0.6408    -0.4378   0.6626
  Ships.SpeedWave.Direction1  -0.0251   0.6940    -0.0362   0.9712
  Ships.SpeedWave.Direction2   0.0705   0.2510     0.2808   0.7795

Residual standard error: 4.747 on 89 degrees of freedom
Multiple R-Squared: 0.1367

       F-statistic: 1.006 on 14 and 89 degrees of freedom, the p-value is 0.4544
```

c. Log Linear Model for MOSIC

```
             *** Linear Model ***

Call: lm(formula = log(Total.Score) ~ (Exposure.Time + Wave.High + Ships.Speed
+ Wave.Direction)^2, data = Mosick.Data.Crew, subset =
       Total.Score > 11.5, na.action = na.exclude)
Residuals:
     Min      1Q Median     3Q    Max
 -0.3538 -0.1502 -0.057 0.1153 0.7254

Coefficients:
                               Value Std. Error t value Pr(>|t|)
                  (Intercept)  1.2585  2.0622     0.6103  0.5432
                Exposure.Time  0.0096  0.0121     0.7962  0.4280
                    Wave.High  0.1777  0.2536     0.7008  0.4853
                  Ships.Speed  0.0240  0.0843     0.2845  0.7767
              Wave.Direction1 -0.0630  0.9626    -0.0654  0.9480
              Wave.Direction2 -0.0260  0.4514    -0.0576  0.9542
     Exposure.Time:Wave.High -0.0015  0.0008    -1.8171  0.0726
   Exposure.Time:Ships.Speed -0.0001  0.0004    -0.3314  0.7411
Exposure.TimeWave.Direction1  0.0021  0.0029     0.7344  0.4646
Exposure.TimeWave.Direction2  0.0007  0.0017     0.4345  0.6650
       Wave.High:Ships.Speed  0.0015  0.0110     0.1385  0.8902
    Wave.HighWave.Direction1 -0.0353  0.0621    -0.5681  0.5714
    Wave.HighWave.Direction2 -0.0201  0.0343    -0.5879  0.5581
  Ships.SpeedWave.Direction1 -0.0013  0.0371    -0.0340  0.9729
  Ships.SpeedWave.Direction2  0.0025  0.0134     0.1874  0.8518

Residual standard error: 0.2538 on 89 degrees of freedom
Multiple R-Squared: 0.1474
F-statistic: 1.099 on 14 and 89 degrees of freedom, the p-value is 0.3697
```

d. Stabilized full Model for MOSIC

```
             *** Linear Model ***
```

```
Call: lm(formula =  - (Total.Score)^-2 ~ (Exposure.Time + Wave.High +
Ships.Speed + Wave.Direction)^2, data = Mosick.Data.Crew, subset =
      Total.Score > 11.5, na.action = na.exclude)
Residuals:
       Min        1Q     Median       3Q      Max
 -0.003094 -0.001254 -0.0002706 0.001291 0.004207

Coefficients:
                              Value Std. Error t value Pr(>|t|)
                (Intercept) -0.0145  0.0147    -0.9889  0.3254
               Exposure.Time  0.0001  0.0001     0.6789  0.4990
                   Wave.High  0.0013  0.0018     0.7346  0.4645
                 Ships.Speed  0.0002  0.0006     0.2888  0.7734
            Wave.Direction1 -0.0009  0.0069    -0.1315  0.8957
            Wave.Direction2  0.0007  0.0032     0.2161  0.8294
     Exposure.Time:Wave.High  0.0000  0.0000    -1.7366  0.0859
   Exposure.Time:Ships.Speed  0.0000  0.0000    -0.2877  0.7742
Exposure.TimeWave.Direction1  0.0000  0.0000     0.7680  0.4445
Exposure.TimeWave.Direction2  0.0000  0.0000     0.2805  0.7797
       Wave.High:Ships.Speed  0.0000  0.0001     0.0634  0.9496
    Wave.HighWave.Direction1 -0.0003  0.0004    -0.7058  0.4821
    Wave.HighWave.Direction2 -0.0002  0.0002    -0.8823  0.3800
  Ships.SpeedWave.Direction1  0.0000  0.0003     0.0681  0.9459
  Ships.SpeedWave.Direction2  0.0000  0.0001    -0.0370  0.9706

Residual standard error: 0.001808 on 89 degrees of freedom
Multiple R-Squared: 0.155
F-statistic: 1.166 on 14 and 89 degrees of freedom, the p-value is 0.3152
```

3. Final Model for MOSIC (Step AIC)

```
      *** Linear Model ***

Call: lm(formula =  - (Total.Score)^-2 ~ Exposure.Time + Wave.High, data =
Mosick.Data.Crew, subset = Total.Score > 11.5, na.action =
      na.exclude)
Residuals:
       Min        1Q     Median       3Q      Max
 -0.003037 -0.001337 -0.0003602 0.001384 0.004359

Coefficients:
               Value Std. Error t value Pr(>|t|)
  (Intercept) -0.0052  0.0008    -6.6145  0.0000
Exposure.Time  0.0000  0.0000    -2.2434  0.0271
    Wave.High  0.0002  0.0001     1.9283  0.0566

Residual standard error: 0.001748 on 101 degrees of freedom
Multiple R-Squared: 0.104
F-statistic: 5.862 on 2 and 101 degrees of freedom, the p-value is 0.003902
```

4. ANOVA for MOSIC Model

```
Analysis of Variance Table

Response:  - (Total.Score)^-2

                          Terms                          Df      RSS
1 (Exp.Time + Wave.High + S.Speed + Wave.Dir)^2          89    0.0002909776
2  Exposure.Time + Wave.High                            101    0.0003085293

Test
   Df      Sum of Sq       F Value      Pr(F)
1
2 -12  -0.00001755172      0.4473722    0.9391408
```

B. MOTION INDUCED TASK INTERRUPTIONS MODELS

1. MITI Length

a. Linear Model for MITI

```
*** Linear Model ***

Call: lm(formula = Length ~ Z.acel + Wave.High + Speed + Wave.Dir + Position,
data = Training.Length.Data, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q   Max
 -26.07 -4.407 -1.584 2.324 67.84

Coefficients:
               Value Std. Error   t value  Pr(>|t|)
(Intercept)  196.0694   99.3001     1.9745    0.0516
     Z.acel   45.9790   11.4034     4.0320    0.0001
  Wave.High  -15.8553   11.8913    -1.3333    0.1861
      Speed   -4.4193    2.7999    -1.5784    0.1183
  Wave.Dir1    1.9663    2.3392     0.8406    0.4030
  Wave.Dir2    5.0573    3.2120     1.5745    0.1192
  Position1   -1.8879    1.6052    -1.1762    0.2429
  Position2    5.6710    1.3629     4.1609    0.0001
  Position3   -1.3406    2.2852    -0.5866    0.5591

Residual standard error: 12.63 on 83 degrees of freedom
Multiple R-Squared: 0.443

F-statistic: 8.251 on 8 and 83 degrees of freedom, the p-value is 3.702e-008
```

b. Stabilize Model for MITI

```
*** Linear Model ***

Call: lm(formula = log(Length) ~ Z.acel + Wave.High + Speed + Wave.Dir +
Position, data = Training.Length.Data, na.action = na.exclude)
Residuals:
   Min      1Q   Median     3Q   Max
```

```
 -1.37 -0.5197 -0.05135 0.4192 1.769

Coefficients:
             Value Std. Error t value Pr(>|t|)
(Intercept)  8.4965  5.4946     1.5463  0.1258
     Z.acel  2.0836  0.6310     3.3021  0.0014
  Wave.High -0.8975  0.6580    -1.3640  0.1762
      Speed -0.0231  0.1549    -0.1492  0.8818
  Wave.Dir1  0.2878  0.1294     2.2235  0.0289
  Wave.Dir2  0.3125  0.1777     1.7583  0.0824
  Position1 -0.2232  0.0888    -2.5131  0.0139
  Position2  0.2601  0.0754     3.4485  0.0009
  Position3 -0.0352  0.1264    -0.2785  0.7814

Residual standard error: 0.6988 on 83 degrees of freedom
Multiple R-Squared: 0.414
F-statistic: 7.331 on 8 and 83 degrees of freedom, the p-value is 2.507e-007
```

c. Final Model for MITI (Step AIC)

```
*** Linear Model ***

Call: lm(formula = log(Length) ~ Z.acel + Wave.High + Wave.Dir + Position, data
= Training.Length.set, na.action = na.exclude)
Residuals:
    Min      1Q   Median     3Q   Max
 -1.349 -0.5171 -0.05089 0.4247 1.781

Coefficients:
             Value Std. Error t value Pr(>|t|)
(Intercept)  8.0784  4.6985     1.7194  0.0892
     Z.acel  2.0595  0.6065     3.3959  0.0010
  Wave.High -0.8967  0.6541    -1.3708  0.1741
  Wave.Dir1  0.3005  0.0971     3.0936  0.0027
  Wave.Dir2  0.3184  0.1722     1.8494  0.0679
  Position1 -0.2264  0.0857    -2.6421  0.0098
  Position2  0.2586  0.0744     3.4782  0.0008
  Position3 -0.0371  0.1251    -0.2966  0.7675

Residual standard error: 0.6947 on 84 degrees of freedom
Multiple R-Squared: 0.4139

F-statistic: 8.473 on 7 and 84 degrees of freedom, the p-value is 8.216e-008
```

2. MITI Frequency

a. Linear Model

```
*** Linear Model ***

Call: lm(formula = Freq ~ Wave.high + Speed + Wave.Dir + t.foil, data =
Freq.Data, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q   Max
 -49.07 -11.72 -1.977 9.874 63.25

Coefficients:
               Value Std. Error   t value  Pr(>|t|)
(Intercept)  170.7662   52.3086     3.2646    0.0021
```

```
  Wave.high  -11.1614    4.5983    -2.4273    0.0194
      Speed   -2.7430    1.9099    -1.4362    0.1580
  Wave.Dir1  -38.8151    4.7620    -8.1510    0.0000
  Wave.Dir2  -14.2540    3.1525    -4.5215    0.0000
     t.foil   -5.2577    6.7073    -0.7839    0.4373

Residual standard error: 21.72 on 44 degrees of freedom
Multiple R-Squared: 0.7257
F-statistic: 23.28 on 5 and 44 degrees of freedom, the p-value is 2.336e-011
```

b. Stabilized Model

```
*** Linear Model ***

Call: lm(formula = sqrt(Freq) ~ Wave.high + Speed + Wave.Dir + t.foil, data =
Freq.Data, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q   Max
 -3.332 -1.677  0.394 1.205 6.833

Coefficients:
              Value Std. Error  t value Pr(>|t|)
(Intercept)  18.4074   5.1005     3.6089   0.0008
  Wave.high  -1.1709   0.4484    -2.6115   0.0123
      Speed  -0.2744   0.1862    -1.4732   0.1478
  Wave.Dir1  -2.9656   0.4643    -6.3868   0.0000
  Wave.Dir2  -1.4051   0.3074    -4.5709   0.0000
     t.foil  -0.6543   0.6540    -1.0005   0.3226

Residual standard error: 2.118 on 44 degrees of freedom
Multiple R-Squared: 0.6638

F-statistic: 17.38 on 5 and 44 degrees of freedom, the p-value is 1.81e-009
```

c. Final Frequency model

```
*** Linear Model ***

Call: lm(formula = sqrt(Freq) ~ Wave.high + Wave.Dir, data = Freq.Data,
na.action = na.exclude)
Residuals:
    Min    1Q  Median    3Q   Max
 -3.231 -1.56 -0.1574 1.314 6.807

Coefficients:
              Value Std. Error  t value Pr(>|t|)
(Intercept)  12.3967   2.9613     4.1862   0.0001
  Wave.high  -1.0845   0.4287    -2.5299   0.0149
  Wave.Dir1  -3.2836   0.4026    -8.1567   0.0000
  Wave.Dir2  -1.6228   0.2752    -5.8978   0.0000

Residual standard error: 2.131 on 46 degrees of freedom
Multiple R-Squared: 0.6442
F-statistic: 27.76 on 3 and 46 degrees of freedom, the p-value is 2.125e-010
```

3. Simulation Results Models

a. Final Models for ER

```
*** Linear Model ***

Call: lm(formula = ERutil ~ MI.freq + MI.length, data = ER.data, na.action =
na.exclude)
Residuals:
      Min        1Q     Median       3Q     Max
 -0.03119 -0.007676 -0.0006048 0.007393 0.06413

Coefficients:
               Value Std. Error   t value  Pr(>|t|)
(Intercept)    0.2736    0.0013   205.8677    0.0000
    MI.freq   -0.0002    0.0000    -7.0961    0.0000
  MI.length    0.0108    0.0042     2.5675    0.0105

Residual standard error: 0.01167 on 647 degrees of freedom
Multiple R-Squared: 0.08111
F-statistic: 28.55 on 2 and 647 degrees of freedom, the p-value is 1.307e-012
```

b. Final Models for NOOW

(a) ASUW

```
*** Linear Model ***

Call: lm(formula = log(NOOWutil) ~ fix.freq + contact.freq + MI.freq +
MI.length + Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg + Pr.Engage,
       data = ASUW.data, na.action = na.exclude)
Residuals:
     Min      1Q  Median     3Q    Max
 -0.2339 -0.0819 -0.0209 0.0801 0.5121

Coefficients:
               Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.4626   0.0373   -12.4084   0.0000
    fix.freq  -0.0170   0.0007   -26.0488   0.0000
contact.freq  -0.0731   0.0013   -54.4978   0.0000
     MI.freq  -0.0026   0.0003    -8.5786   0.0000
   MI.length   0.0837   0.0177     4.7385   0.0000
Crs.Chg.freq  -0.0093   0.0017    -5.3165   0.0000
Spd.Chg.freq  -0.0166   0.0017    -9.4896   0.0000
  Pr.Crs.Chg   0.2029   0.0393     5.1659   0.0000
  Pr.Spd.Chg   0.2091   0.0466     4.4917   0.0000
   Pr.Engage   0.3943   0.0196    20.1149   0.0000

Residual standard error: 0.1319 on 640 degrees of freedom
Multiple R-Squared: 0.8705
F-statistic: 478.1 on 9 and 640 degrees of freedom, the p-value is 0
```

(b) NSFS

```
*** Linear Model ***

Call: lm(formula = 1/(NOOWutil) ~ fix.freq + contact.freq + MI.freq + MI.length
+ Crs.Chg.freq + Spd.Chg.freq + Strike.freq + Pr.Crs.Chg, data =
       NSFS.data, na.action = na.exclude)
Residuals:
   Min     1Q Median     3Q   Max
 -2.43 -0.424 0.1043 0.4447 1.362

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -1.3015   0.1630    -7.9829   0.0000
    fix.freq   0.2573   0.0077    33.5840   0.0000
contact.freq   0.0386   0.0023    16.7066   0.0000
     MI.freq   0.0105   0.0016     6.7394   0.0000
   MI.length  -0.1326   0.0924    -1.4356   0.1516
Crs.Chg.freq   0.0227   0.0026     8.5846   0.0000
Spd.Chg.freq   0.0179   0.0017    10.6758   0.0000
 Strike.freq   0.1124   0.0033    34.1836   0.0000
  Pr.Crs.Chg  -0.5385   0.2055    -2.6210   0.0090

Residual standard error: 0.6902 on 641 degrees of freedom
Multiple R-Squared: 0.8161
F-statistic: 355.6 on 8 and 641 degrees of freedom, the p-value is 0
```

(c) Littoral Transit

```
*** Linear Model ***

Call: lm(formula = 1/(NOOWutil) ~ fix.freq + contac.freq + MI.freq + MI.length
+ Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg, data = LT.data,
       na.action = na.exclude)
Residuals:
    Min      1Q  Median     3Q   Max
 -1.983 -0.5589 0.01522 0.5638 2.251

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.1592   0.1715    -0.9283   0.3536
    fix.freq   0.1330   0.0038    34.9483   0.0000
 contac.freq   0.4312   0.0078    55.0888   0.0000
     MI.freq   0.0142   0.0017     8.1391   0.0000
   MI.length  -0.4082   0.1031    -3.9612   0.0001
Crs.Chg.freq   0.0322   0.0029    10.9403   0.0000
Spd.Chg.freq   0.0222   0.0019    11.8778   0.0000
  Pr.Crs.Chg  -1.6148   0.1082   -14.9305   0.0000
  Pr.Spd.Chg  -1.6952   0.1214   -13.9652   0.0000

Residual standard error: 0.7697 on 641 degrees of freedom
Multiple R-Squared: 0.8876
F-statistic: 632.5 on 8 and 641 degrees of freedom, the p-value is 0
```

(d) Open Transit

```
*** Linear Model ***

Call: lm(formula = log(NOOWutil) ~ fix.freq + contact.freq + MI.freq +
MI.length + Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg, data =
OT.data,
       na.action = na.exclude)
Residuals:
     Min      1Q    Median      3Q    Max
 -0.2151 -0.0749 -0.007476 0.05054 0.5443

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -2.2870   0.0283   -80.9161   0.0000
    fix.freq  -0.0166   0.0003   -51.5037   0.0000
contact.freq  -0.0056   0.0001   -40.3005   0.0000
     MI.freq  -0.0027   0.0002   -11.2297   0.0000
   MI.length   0.0898   0.0145     6.2025   0.0000
Crs.Chg.freq  -0.0004   0.0001    -4.7526   0.0000
Spd.Chg.freq  -0.0006   0.0001    -6.9163   0.0000
  Pr.Crs.Chg   0.1756   0.0322     5.4555   0.0000
  Pr.Spd.Chg   0.2588   0.0382     6.7803   0.0000

Residual standard error: 0.1082 on 641 degrees of freedom
Multiple R-Squared: 0.8777
F-statistic: 575.2 on 8 and 641 degrees of freedom, the p-value is 0
```

c. Final Models for OOD

(a) ASUW

```
*** Linear Model ***

Call: lm(formula = 1/(OODutil) ~ contact.freq + MI.freq + MI.length +
Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg + Pr.Engage, data =
       OOD.ASUW.data, na.action = na.exclude)
Residuals:
     Min     1Q    Median     3Q   Max
 -0.8961 -0.328 -0.002059 0.2571 2.381

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)   1.7134   0.1241    13.8022   0.0000
contact.freq   0.3652   0.0047    78.4078   0.0000
     MI.freq   0.0105   0.0010    10.1065   0.0000
   MI.length  -0.2406   0.0613    -3.9246   0.0001
Crs.Chg.freq   0.1045   0.0061    17.2506   0.0000
Spd.Chg.freq   0.0427   0.0061     7.0434   0.0000
  Pr.Crs.Chg  -3.0412   0.1363   -22.3151   0.0000
  Pr.Spd.Chg  -2.1920   0.1616   -13.5671   0.0000
   Pr.Engage  -1.6165   0.0680   -23.7617   0.0000

Residual standard error: 0.4579 on 641 degrees of freedom
Multiple R-Squared: 0.9247
F-statistic: 984.5 on 8 and 641 degrees of freedom, the p-value is 0
```

(b) NSFS

```
*** Linear Model ***

Call: lm(formula = 1/(OODutil) ~ fix.freq + contact.freq + MI.freq + MI.length
+ Crs.Chg.freq + Spd.Chg.freq + Strike.freq + Pr.Crs.Chg + Pr.Spd.Chg,
       data = NSFS.data, na.action = na.exclude)
Residuals:
    Min      1Q Median     3Q   Max
 -4.126 -0.5267 0.1459 0.7092 2.376

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.9408   0.2745    -3.4278   0.0006
    fix.freq   0.0926   0.0124     7.4569   0.0000
contact.freq   0.1159   0.0037    30.9318   0.0000
     MI.freq   0.0154   0.0025     6.0980   0.0000
   MI.length  -0.1916   0.1498    -1.2792   0.2013
Crs.Chg.freq   0.0671   0.0043    15.6797   0.0000
Spd.Chg.freq   0.0200   0.0027     7.3629   0.0000
 Strike.freq   0.1949   0.0053    36.5404   0.0000
  Pr.Crs.Chg  -3.3672   0.3331   -10.1094   0.0000
  Pr.Spd.Chg  -1.7302   0.3947    -4.3830   0.0000

Residual standard error: 1.119 on 640 degrees of freedom
Multiple R-Squared: 0.8153
F-statistic: 313.8 on 9 and 640 degrees of freedom, the p-value is 0
```

(c) Littoral Transit

```
*** Linear Model ***

Call: lm(formula = 1/(OODutil) ~ fix.freq + contac.freq + MI.freq + MI.length +
Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg, data = LT.data,
       na.action = na.exclude)
Residuals:
    Min      1Q  Median     3Q   Max
 -1.794 -0.3927 -0.1032 0.2799 3.963

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)   1.5280   0.1705     8.9603   0.0000
    fix.freq   0.0283   0.0038     7.4743   0.0000
 contac.freq   0.4989   0.0078    64.0966   0.0000
     MI.freq   0.0126   0.0017     7.2433   0.0000
   MI.length  -0.2654   0.1025    -2.5893   0.0098
Crs.Chg.freq   0.0414   0.0029    14.1366   0.0000
Spd.Chg.freq   0.0086   0.0019     4.6205   0.0000
  Pr.Crs.Chg  -3.0439   0.1076   -28.3003   0.0000
  Pr.Spd.Chg  -2.4582   0.1207   -20.3646   0.0000

Residual standard error: 0.7654 on 641 degrees of freedom
Multiple R-Squared: 0.8988
F-statistic: 711.5 on 8 and 641 degrees of freedom, the p-value is 0
```

(d) Open Transit

```
*** Linear Model ***

Call: lm(formula = 1/(OODutil) ~ fix.freq + contact.freq + MI.freq + MI.length
+ Crs.Chg.freq + Spd.Chg.freq + Pr.Crs.Chg + Pr.Spd.Chg, data = OT.data,
       na.action = na.exclude)
Residuals:
    Min     1Q   Median    3Q   Max
 -15.78 -2.703 0.002482 2.881 23.52

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)   8.0008   1.3152     6.0832   0.0000
    fix.freq   0.2123   0.0150    14.1694   0.0000
contact.freq   0.4795   0.0064    74.7329   0.0000
     MI.freq   0.1205   0.0114    10.5785   0.0000
   MI.length  -3.3716   0.6739    -5.0031   0.0000
Crs.Chg.freq   0.0588   0.0037    15.7078   0.0000
Spd.Chg.freq   0.0229   0.0037     6.1163   0.0000
  Pr.Crs.Chg -40.1795   1.4983   -26.8171   0.0000
  Pr.Spd.Chg -29.4472   1.7759   -16.5817   0.0000

Residual standard error: 5.034 on 641 degrees of freedom
Multiple R-Squared: 0.9186
F-statistic: 903.6 on 8 and 641 degrees of freedom, the p-value is 0
```

d. Final Model for TAO

(e) ASUW

```
*** Linear Model ***

Call: lm(formula = log(TAOutil) ~ contact.freq + MI.freq + MI.length +
Radio.freq + Pr.Engage, data = ASUW.data, na.action = na.exclude)
Residuals:
     Min      1Q   Median      3Q    Max
 -0.3239 -0.1156 -0.01703 0.08557 0.5329

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.6471   0.0289   -22.3853   0.0000
contact.freq  -0.1072   0.0016   -68.1886   0.0000
     MI.freq  -0.0020   0.0003    -5.6230   0.0000
   MI.length   0.0518   0.0207     2.5043   0.0125
  Radio.freq  -0.0122   0.0008   -14.7209   0.0000
   Pr.Engage   0.9937   0.0230    43.2773   0.0000

Residual standard error: 0.1546 on 644 degrees of freedom
Multiple R-Squared: 0.9137
F-statistic: 1364 on 5 and 644 degrees of freedom, the p-value is 0
```

(f) NSFS

```
*** Linear Model ***

Call: lm(formula = 1/(TAOutil) ~ contact.freq + MI.freq + MI.length +
Strike.freq + Radio.freq, data = NSFS.data, na.action = na.exclude)
Residuals:
    Min      1Q  Median     3Q   Max
 -1.273 -0.2994 0.08169 0.3371 1.042
```

```
Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.3109   0.0834    -3.7280   0.0002
contact.freq   0.0210   0.0015    14.1029   0.0000
     MI.freq   0.0077   0.0010     7.6124   0.0000
   MI.length  -0.1983   0.0597    -3.3227   0.0009
 Strike.freq   0.1552   0.0021    73.0223   0.0000
  Radio.freq   0.0356   0.0024    14.9283   0.0000

Residual standard error: 0.4459 on 644 degrees of freedom
Multiple R-Squared: 0.8997
F-statistic: 1156 on 5 and 644 degrees of freedom, the p-value is 0
```

(g) Littoral Transit

```
*** Linear Model ***

Call: lm(formula = 1/(TAOutil) ~ fix.freq + contac.freq + MI.freq + MI.length +
Radio.freq, data = LT.data, na.action = na.exclude)
Residuals:
    Min     1Q Median     3Q   Max
 -2.955 -0.536 0.1775 0.7421 1.798

Coefficients:
               Value Std. Error  t value Pr(>|t|)
(Intercept)  -1.4394   0.1851    -7.7769   0.0000
   fix.freq   0.0432   0.0049     8.8612   0.0000
contac.freq   0.5723   0.0100    57.0639   0.0000
    MI.freq   0.0187   0.0022     8.3802   0.0000
  MI.length  -0.3355   0.1320    -2.5405   0.0113
 Radio.freq   0.1765   0.0053    33.4323   0.0000

Residual standard error: 0.9863 on 644 degrees of freedom
Multiple R-Squared: 0.8784
F-statistic: 930.3 on 5 and 644 degrees of freedom, the p-value is 0
```

(h) Open Transit

```
*** Linear Model ***

Call: lm(formula = 1/(TAOutil) ~ fix.freq + contact.freq + MI.freq + MI.length
+ Radio.freq, data = OT.data, na.action = na.exclude)
Residuals:
    Min     1Q Median    3Q   Max
 -8.564 -1.957 0.3073 2.261 9.759

Coefficients:
                Value Std. Error  t value Pr(>|t|)
 (Intercept)  -0.7419   0.8673    -0.8554   0.3926
    fix.freq   0.0991   0.0096    10.3671   0.0000
contact.freq   0.1790   0.0041    43.7347   0.0000
     MI.freq   0.0740   0.0073    10.1826   0.0000
   MI.length  -2.4055   0.4299    -5.5957   0.0000
  Radio.freq   0.3476   0.0143    24.3772   0.0000

Residual standard error: 3.211 on 644 degrees of freedom
Multiple R-Squared: 0.8082
F-statistic: 542.6 on 5 and 644 degrees of freedom, the p-value is 0
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. OPERATIONS MANUAL TABLE

The Operations Manual gives an interval of the lower bound of expected levels of motion sickness. To determine the correct lower bound interval, first locate the maximum wave height that the vessel will encounter enroute to its destination port in the "Wave Height" column. Next, locate the time that the landing force personnel will be at sea in the "Exp. Time" column that matches with the correct wave height. The next two columns in that row indicate the lower and upper bounds of a 95% PI of the lower bound of MSAQ combined scores that landing force personnel will have upon arrival. The last two columns of that row indicate the lower and upper bounds of a 95% PI of the lower bound of motion sickness levels that landing force personnel will have upon arrival.

Exp. Time (Hrs.)	Wave Height (ft.)	LB (MSAQ combined score)	UB (MSAQ combined score)	Motion Sick Levels Upper/Lower Bound	
0	0	10.5560977	17.1364456	Moderate	Moderate
1	0	10.5514455	17.1140524	Moderate	Moderate
2	0	10.5467878	17.0917438	Moderate	Moderate
3	0	10.5421246	17.0695193	Moderate	Moderate
4	0	10.5374561	17.0473785	Moderate	Moderate
5	0	10.5327820	17.0253211	Moderate	Moderate
6	0	10.5281026	17.0033467	Moderate	Moderate
7	0	10.5234178	16.9814549	Moderate	Moderate
8	0	10.5187276	16.9596452	Moderate	Moderate
9	0	10.5140321	16.9379174	Moderate	Moderate
10	0	10.5093312	16.9162710	Moderate	Moderate
11	0	10.5046249	16.8947057	Moderate	Moderate
12	0	10.4999133	16.8732210	Moderate	Moderate
13	0	10.4951964	16.8518167	Moderate	Moderate
14	0	10.4904742	16.8304923	Moderate	Moderate
15	0	10.4857467	16.8092475	Moderate	Moderate
16	0	10.4810139	16.7880819	Moderate	Moderate
17	0	10.4762759	16.7669951	Moderate	Moderate
18	0	10.4715326	16.7459868	Moderate	Moderate
19	0	10.4667841	16.7250567	Moderate	Moderate
20	0	10.4620304	16.7042043	Moderate	Moderate
21	0	10.4572714	16.6834292	Moderate	Moderate
22	0	10.4525073	16.6627313	Moderate	Moderate
23	0	10.4477380	16.6421100	Moderate	Moderate
24	0	10.4429636	16.6215651	Moderate	Moderate
25	0	10.4381840	16.6010962	Moderate	Moderate

26	0	10.4333993	16.5807030	Moderate	Moderate
27	0	10.4286094	16.5603850	Moderate	Moderate
28	0	10.4238145	16.5401421	Moderate	Moderate
29	0	10.4190145	16.5199737	Moderate	Moderate
30	0	10.4142095	16.4998797	Moderate	Moderate
31	0	10.4093994	16.4798596	Moderate	Moderate
32	0	10.4045843	16.4599131	Moderate	Moderate
33	0	10.3997642	16.4400399	Moderate	Moderate
34	0	10.3949391	16.4202397	Moderate	Moderate
35	0	10.3901090	16.4005121	Moderate	Moderate
36	0	10.3852740	16.3808567	Moderate	Moderate
37	0	10.3804340	16.3612734	Moderate	Moderate
38	0	10.3755892	16.3417617	Moderate	Moderate
39	0	10.3707394	16.3223214	Moderate	Moderate
40	0	10.3658848	16.3029520	Moderate	Moderate
41	0	10.3610253	16.2836533	Moderate	Moderate
42	0	10.3561609	16.2644250	Moderate	Moderate
43	0	10.3512918	16.2452668	Moderate	Moderate
44	0	10.3464178	16.2261783	Moderate	Moderate
45	0	10.3415391	16.2071593	Moderate	Moderate
46	0	10.3366556	16.1882093	Moderate	Moderate
47	0	10.3317673	16.1693282	Moderate	Moderate
48	0	10.3268744	16.1505156	Moderate	Moderate
0	1	10.7362857	17.5666145	Moderate	Moderate
1	1	10.7313340	17.5426946	Moderate	Moderate
2	1	10.7263766	17.5188686	Moderate	Moderate
3	1	10.7214136	17.4951360	Moderate	Moderate
4	1	10.7164449	17.4714963	Moderate	Moderate
5	1	10.7114706	17.4479491	Moderate	Moderate
6	1	10.7064907	17.4244939	Moderate	Moderate
7	1	10.7015051	17.4011303	Moderate	Moderate
8	1	10.6965140	17.3778578	Moderate	Moderate
9	1	10.6915173	17.3546760	Moderate	Moderate
10	1	10.6865150	17.3315844	Moderate	Moderate
11	1	10.6815072	17.3085827	Moderate	Moderate
12	1	10.6764939	17.2856703	Moderate	Moderate
13	1	10.6714750	17.2628468	Moderate	Moderate
14	1	10.6664506	17.2401119	Moderate	Moderate
15	1	10.6614208	17.2174650	Moderate	Moderate
16	1	10.6563855	17.1949057	Moderate	Moderate
17	1	10.6513448	17.1724337	Moderate	Moderate
18	1	10.6462986	17.1500485	Moderate	Moderate
19	1	10.6412470	17.1277496	Moderate	Moderate
20	1	10.6361900	17.1055368	Moderate	Moderate
21	1	10.6311277	17.0834095	Moderate	Moderate
22	1	10.6260600	17.0613673	Moderate	Moderate
23	1	10.6209869	17.0394099	Moderate	Moderate
24	1	10.6159086	17.0175368	Moderate	Moderate
25	1	10.6108249	16.9957476	Moderate	Moderate
26	1	10.6057359	16.9740420	Moderate	Moderate
27	1	10.6006417	16.9524195	Moderate	Moderate

28	1	10.5955423	16.9308797	Moderate	Moderate
29	1	10.5904376	16.9094223	Moderate	Moderate
30	1	10.5853277	16.8880468	Moderate	Moderate
31	1	10.5802127	16.8667529	Moderate	Moderate
32	1	10.5750924	16.8455401	Moderate	Moderate
33	1	10.5699671	16.8244082	Moderate	Moderate
34	1	10.5648366	16.8033566	Moderate	Moderate
35	1	10.5597010	16.7823851	Moderate	Moderate
36	1	10.5545603	16.7614932	Moderate	Moderate
37	1	10.5494146	16.7406806	Moderate	Moderate
38	1	10.5442639	16.7199469	Moderate	Moderate
39	1	10.5391081	16.6992918	Moderate	Moderate
40	1	10.5339474	16.6787147	Moderate	Moderate
41	1	10.5287816	16.6582155	Moderate	Moderate
42	1	10.5236110	16.6377937	Moderate	Moderate
43	1	10.5184354	16.6174489	Moderate	Moderate
44	1	10.5132549	16.5971809	Moderate	Moderate
45	1	10.5080695	16.5769892	Moderate	Moderate
46	1	10.5028793	16.5568735	Moderate	Moderate
47	1	10.4976842	16.5368334	Moderate	Moderate
48	1	10.4924844	16.5168686	Moderate	Moderate
0	2	10.9178456	18.0376337	Moderate	Moderate
1	2	10.9125712	18.0119953	Moderate	Moderate
2	2	10.9072909	17.9864616	Moderate	Moderate
3	2	10.9020047	17.9610320	Moderate	Moderate
4	2	10.8967128	17.9357060	Moderate	Moderate
5	2	10.8914150	17.9104830	Moderate	Moderate
6	2	10.8861115	17.8853625	Moderate	Moderate
7	2	10.8808022	17.8603439	Moderate	Moderate
8	2	10.8754872	17.8354268	Moderate	Moderate
9	2	10.8701665	17.8106106	Moderate	Moderate
10	2	10.8648401	17.7858948	Moderate	Moderate
11	2	10.8595080	17.7612789	Moderate	Moderate
12	2	10.8541702	17.7367623	Moderate	Moderate
13	2	10.8488268	17.7123447	Moderate	Moderate
14	2	10.8434777	17.6880254	Moderate	Moderate
15	2	10.8381231	17.6638039	Moderate	Moderate
16	2	10.8327629	17.6396799	Moderate	Moderate
17	2	10.8273971	17.6156527	Moderate	Moderate
18	2	10.8220258	17.5917219	Moderate	Moderate
19	2	10.8166490	17.5678870	Moderate	Moderate
20	2	10.8112666	17.5441475	Moderate	Moderate
21	2	10.8058788	17.5205029	Moderate	Moderate
22	2	10.8004855	17.4969528	Moderate	Moderate
23	2	10.7950868	17.4734966	Moderate	Moderate
24	2	10.7896827	17.4501340	Moderate	Moderate
25	2	10.7842732	17.4268644	Moderate	Moderate
26	2	10.7788583	17.4036873	Moderate	Moderate
27	2	10.7734381	17.3806024	Moderate	Moderate
28	2	10.7680126	17.3576091	Moderate	Moderate
29	2	10.7625817	17.3347070	Moderate	Moderate

30	2	10.7571456	17.3118956	Moderate	Moderate
31	2	10.7517043	17.2891744	Moderate	Moderate
32	2	10.7462577	17.2665431	Moderate	Moderate
33	2	10.7408059	17.2440012	Moderate	Moderate
34	2	10.7353489	17.2215482	Moderate	Moderate
35	2	10.7298868	17.1991837	Moderate	Moderate
36	2	10.7244196	17.1769073	Moderate	Moderate
37	2	10.7189472	17.1547185	Moderate	Moderate
38	2	10.7134698	17.1326168	Moderate	Moderate
39	2	10.7079873	17.1106019	Moderate	Moderate
40	2	10.7024999	17.0886734	Moderate	Moderate
41	2	10.6970074	17.0668307	Moderate	Moderate
42	2	10.6915099	17.0450735	Moderate	Moderate
43	2	10.6860075	17.0234014	Moderate	Moderate
44	2	10.6805001	17.0018139	Moderate	Moderate
45	2	10.6749879	16.9803106	Moderate	Moderate
46	2	10.6694708	16.9588911	Moderate	Moderate
47	2	10.6639489	16.9375550	Moderate	Moderate
48	2	10.6584222	16.9163019	Moderate	Moderate
0	3	11.0998810	18.5555038	Moderate	Moderate
1	3	11.0942600	18.5279183	Moderate	Moderate
2	3	11.0886331	18.5004501	Moderate	Moderate
3	3	11.0830002	18.4730984	Moderate	Moderate
4	3	11.0773615	18.4458627	Moderate	Moderate
5	3	11.0717170	18.4187423	Moderate	Moderate
6	3	11.0660665	18.3917365	Moderate	Moderate
7	3	11.0604103	18.3648448	Moderate	Moderate
8	3	11.0547483	18.3380666	Moderate	Moderate
9	3	11.0490805	18.3114012	Moderate	Moderate
10	3	11.0434070	18.2848480	Moderate	Moderate
11	3	11.0377277	18.2584064	Moderate	Moderate
12	3	11.0320427	18.2320759	Moderate	Moderate
13	3	11.0263521	18.2058557	Moderate	Moderate
14	3	11.0206558	18.1797454	Moderate	Moderate
15	3	11.0149538	18.1537443	Moderate	Moderate
16	3	11.0092463	18.1278519	Moderate	Moderate
17	3	11.0035331	18.1020675	Moderate	Moderate
18	3	10.9978144	18.0763906	Moderate	Moderate
19	3	10.9920902	18.0508206	Moderate	Moderate
20	3	10.9863604	18.0253570	Moderate	Moderate
21	3	10.9806252	17.9999991	Moderate	Moderate
22	3	10.9748845	17.9747465	Moderate	Moderate
23	3	10.9691384	17.9495985	Moderate	Moderate
24	3	10.9633868	17.9245546	Moderate	Moderate
25	3	10.9576299	17.8996143	Moderate	Moderate
26	3	10.9518676	17.8747770	Moderate	Moderate
27	3	10.9461000	17.8500422	Moderate	Moderate
28	3	10.9403271	17.8254093	Moderate	Moderate
29	3	10.9345489	17.8008777	Moderate	Moderate
30	3	10.9287655	17.7764471	Moderate	Moderate
31	3	10.9229769	17.7521167	Moderate	Moderate

32	3	10.9171830	17.7278862	Moderate	Moderate
33	3	10.9113840	17.7037549	Moderate	Moderate
34	3	10.9055799	17.6797224	Moderate	Moderate
35	3	10.8997706	17.6557881	Moderate	Moderate
36	3	10.8939563	17.6319516	Moderate	Moderate
37	3	10.8881369	17.6082123	Moderate	Moderate
38	3	10.8823125	17.5845697	Moderate	Moderate
39	3	10.8764831	17.5610233	Moderate	Moderate
40	3	10.8706488	17.5375726	Moderate	Moderate
41	3	10.8648095	17.5142172	Moderate	Moderate
42	3	10.8589654	17.4909564	Moderate	Moderate
43	3	10.8531163	17.4677899	Moderate	Moderate
44	3	10.8472625	17.4447172	Moderate	Moderate
45	3	10.8414038	17.4217377	Moderate	Moderate
46	3	10.8355403	17.3988509	Moderate	Moderate
47	3	10.8296721	17.3760565	Moderate	Moderate
48	3	10.8237992	17.3533539	Moderate	Moderate
0	4	11.2814180	19.1272101	Moderate	Moderate
1	4	11.2754270	19.0974027	Moderate	Moderate
2	4	11.2694300	19.0677273	Moderate	Moderate
3	4	11.2634272	19.0381831	Moderate	Moderate
4	4	11.2574185	19.0087694	Moderate	Moderate
5	4	11.2514040	18.9794855	Moderate	Moderate
6	4	11.2453837	18.9503306	Moderate	Moderate
7	4	11.2393577	18.9213040	Moderate	Moderate
8	4	11.2333259	18.8924048	Moderate	Moderate
9	4	11.2272884	18.8636325	Moderate	Moderate
10	4	11.2212452	18.8349863	Moderate	Moderate
11	4	11.2151964	18.8064654	Moderate	Moderate
12	4	11.2091419	18.7780692	Moderate	Moderate
13	4	11.2030818	18.7497968	Moderate	Moderate
14	4	11.1970162	18.7216478	Moderate	Moderate
15	4	11.1909450	18.6936212	Moderate	Moderate
16	4	11.1848682	18.6657165	Moderate	Moderate
17	4	11.1787860	18.6379329	Moderate	Moderate
18	4	11.1726983	18.6102698	Moderate	Moderate
19	4	11.1666052	18.5827265	Moderate	Moderate
20	4	11.1605067	18.5553023	Moderate	Moderate
21	4	11.1544028	18.5279966	Moderate	Moderate
22	4	11.1482935	18.5008086	Moderate	Moderate
23	4	11.1421790	18.4737378	Moderate	Moderate
24	4	11.1360591	18.4467834	Moderate	Moderate
25	4	11.1299340	18.4199449	Moderate	Moderate
26	4	11.1238036	18.3932215	Moderate	Moderate
27	4	11.1176681	18.3666126	Moderate	Moderate
28	4	11.1115274	18.3401177	Moderate	Moderate
29	4	11.1053816	18.3137360	Moderate	Moderate
30	4	11.0992306	18.2874670	Moderate	Moderate
31	4	11.0930746	18.2613099	Moderate	Moderate
32	4	11.0869136	18.2352643	Moderate	Moderate
33	4	11.0807475	18.2093295	Moderate	Moderate

34	4	11.0745765	18.1835048	Moderate	Moderate
35	4	11.0684006	18.1577896	Moderate	Moderate
36	4	11.0622197	18.1321835	Moderate	Moderate
37	4	11.0560340	18.1066857	Moderate	Moderate
38	4	11.0498434	18.0812957	Moderate	Moderate
39	4	11.0436481	18.0560128	Moderate	Moderate
40	4	11.0374479	18.0308365	Moderate	Moderate
41	4	11.0312430	18.0057663	Moderate	Moderate
42	4	11.0250335	17.9808015	Moderate	Moderate
43	4	11.0188192	17.9559415	Moderate	Moderate
44	4	11.0126004	17.9311859	Moderate	Moderate
45	4	11.0063769	17.9065339	Moderate	Moderate
46	4	11.0001489	17.8819851	Moderate	Moderate
47	4	10.9939163	17.8575389	Moderate	Moderate
48	4	10.9876793	17.8331947	Moderate	Moderate
0	5	11.4614370	19.7609104	Moderate	Moderate
1	5	11.4550532	19.7285480	Moderate	Moderate
2	5	11.4486637	19.6963351	Moderate	Moderate
3	5	11.4422685	19.6642710	Moderate	Moderate
4	5	11.4358676	19.6323546	Moderate	Moderate
5	5	11.4294611	19.6005851	Moderate	Moderate
6	5	11.4230490	19.5689616	Moderate	Moderate
7	5	11.4166313	19.5374833	Moderate	Moderate
8	5	11.4102080	19.5061491	Moderate	Moderate
9	5	11.4037793	19.4749583	Moderate	Moderate
10	5	11.3973451	19.4439099	Moderate	Moderate
11	5	11.3909054	19.4130032	Moderate	Moderate
12	5	11.3844603	19.3822372	Moderate	Moderate
13	5	11.3780098	19.3516112	Moderate	Moderate
14	5	11.3715539	19.3211242	Moderate	Moderate
15	5	11.3650927	19.2907755	Moderate	Moderate
16	5	11.3586263	19.2605642	Moderate	Moderate
17	5	11.3521545	19.2304895	Moderate	Moderate
18	5	11.3456775	19.2005506	Moderate	Moderate
19	5	11.3391954	19.1707466	Moderate	Moderate
20	5	11.3327081	19.1410768	Moderate	Moderate
21	5	11.3262156	19.1115404	Moderate	Moderate
22	5	11.3197181	19.0821366	Moderate	Moderate
23	5	11.3132155	19.0528645	Moderate	Moderate
24	5	11.3067079	19.0237235	Moderate	Moderate
25	5	11.3001952	18.9947127	Moderate	Moderate
26	5	11.2936777	18.9658314	Moderate	Moderate
27	5	11.2871552	18.9370788	Moderate	Moderate
28	5	11.2806278	18.9084542	Moderate	Moderate
29	5	11.2740956	18.8799568	Moderate	Moderate
30	5	11.2675586	18.8515859	Moderate	Moderate
31	5	11.2610168	18.8233406	Moderate	Moderate
32	5	11.2544702	18.7952204	Moderate	Moderate
33	5	11.2479190	18.7672245	Moderate	Moderate
34	5	11.2413631	18.7393521	Moderate	Moderate
35	5	11.2348026	18.7116025	Moderate	Moderate

36	5	11.2282375	18.6839750	Moderate	Moderate
37	5	11.2216678	18.6564690	Moderate	Moderate
38	5	11.2150936	18.6290836	Moderate	Moderate
39	5	11.2085149	18.6018183	Moderate	Moderate
40	5	11.2019318	18.5746722	Moderate	Moderate
41	5	11.1953444	18.5476448	Moderate	Moderate
42	5	11.1887525	18.5207354	Moderate	Moderate
43	5	11.1821563	18.4939432	Moderate	Moderate
44	5	11.1755559	18.4672676	Moderate	Moderate
45	5	11.1689512	18.4407079	Moderate	Moderate
46	5	11.1623423	18.4142635	Moderate	Moderate
47	5	11.1557293	18.3879337	Moderate	Moderate
48	5	11.1491121	18.3617178	Moderate	Moderate
0	6	11.6389096	20.4661842	Moderate	Moderate
1	6	11.6321120	20.4308598	Moderate	Moderate
2	6	11.6253091	20.3957061	Moderate	Moderate
3	6	11.6185008	20.3607221	Moderate	Moderate
4	6	11.6116872	20.3259066	Moderate	Moderate
5	6	11.6048682	20.2912586	Moderate	Moderate
6	6	11.5980440	20.2567769	Moderate	Moderate
7	6	11.5912146	20.2224605	Moderate	Moderate
8	6	11.5843800	20.1883083	Moderate	Moderate
9	6	11.5775403	20.1543193	Moderate	Moderate
10	6	11.5706954	20.1204924	Moderate	Moderate
11	6	11.5638455	20.0868266	Moderate	Moderate
12	6	11.5569905	20.0533208	Moderate	Moderate
13	6	11.5501304	20.0199740	Moderate	Moderate
14	6	11.5432655	19.9867853	Moderate	Moderate
15	6	11.5363955	19.9537535	Moderate	Moderate
16	6	11.5295207	19.9208778	Moderate	Moderate
17	6	11.5226410	19.8881571	Moderate	Moderate
18	6	11.5157565	19.8555904	Moderate	Moderate
19	6	11.5088672	19.8231768	Moderate	Moderate
20	6	11.5019731	19.7909153	Moderate	Moderate
21	6	11.4950743	19.7588049	Moderate	Moderate
22	6	11.4881708	19.7268447	Moderate	Moderate
23	6	11.4812628	19.6950338	Moderate	Moderate
24	6	11.4743501	19.6633712	Moderate	Moderate
25	6	11.4674328	19.6318560	Moderate	Moderate
26	6	11.4605110	19.6004873	Moderate	Moderate
27	6	11.4535848	19.5692642	Moderate	Moderate
28	6	11.4466541	19.5381857	Moderate	Moderate
29	6	11.4397190	19.5072510	Moderate	Moderate
30	6	11.4327796	19.4764591	Moderate	Moderate
31	6	11.4258358	19.4458092	Moderate	Moderate
32	6	11.4188878	19.4153005	Moderate	Moderate
33	6	11.4119355	19.3849320	Moderate	Moderate
34	6	11.4049790	19.3547029	Moderate	Moderate
35	6	11.3980184	19.3246123	Moderate	Moderate
36	6	11.3910537	19.2946594	Moderate	Moderate
37	6	11.3840849	19.2648433	Moderate	Moderate

38	6	11.3771121	19.2351632	Moderate	Moderate
39	6	11.3701352	19.2056183	Moderate	Moderate
40	6	11.3631545	19.1762077	Moderate	Moderate
41	6	11.3561699	19.1469307	Moderate	Moderate
42	6	11.3491814	19.1177863	Moderate	Moderate
43	6	11.3421891	19.0887739	Moderate	Moderate
44	6	11.3351930	19.0598926	Moderate	Moderate
45	6	11.3281932	19.0311416	Moderate	Moderate
46	6	11.3211897	19.0025202	Moderate	Moderate
47	6	11.3141826	18.9740275	Moderate	Moderate
48	6	11.3071719	18.9456628	Moderate	Moderate
0	7	11.8128396	21.2543734	Moderate	Moderate
1	7	11.8056096	21.2155854	Moderate	Moderate
2	7	11.7983747	21.1769939	Moderate	Moderate
3	7	11.7911350	21.1385974	Moderate	Moderate
4	7	11.7838905	21.1003948	Moderate	Moderate
5	7	11.7766412	21.0623846	Moderate	Moderate
6	7	11.7693871	21.0245654	Moderate	Moderate
7	7	11.7621284	20.9869360	Moderate	Moderate
8	7	11.7548651	20.9494952	Moderate	Moderate
9	7	11.7475971	20.9122414	Moderate	Moderate
10	7	11.7403245	20.8751737	Moderate	Moderate
11	7	11.7330474	20.8382905	Moderate	Moderate
12	7	11.7257658	20.8015907	Moderate	Moderate
13	7	11.7184798	20.7650731	Moderate	Moderate
14	7	11.7111894	20.7287364	Moderate	Moderate
15	7	11.7038945	20.6925793	Moderate	Moderate
16	7	11.6965954	20.6566007	Moderate	Moderate
17	7	11.6892919	20.6207994	Moderate	Moderate
18	7	11.6819843	20.5851742	Moderate	Moderate
19	7	11.6746724	20.5497239	Moderate	Moderate
20	7	11.6673563	20.5144473	Moderate	Moderate
21	7	11.6600361	20.4793432	Moderate	Moderate
22	7	11.6527119	20.4444106	Moderate	Moderate
23	7	11.6453836	20.4096482	Moderate	Moderate
24	7	11.6380513	20.3750550	Moderate	Moderate
25	7	11.6307151	20.3406299	Moderate	Moderate
26	7	11.6233749	20.3063716	Moderate	Moderate
27	7	11.6160309	20.2722791	Moderate	Moderate
28	7	11.6086831	20.2383514	Moderate	Moderate
29	7	11.6013315	20.2045873	Moderate	Moderate
30	7	11.5939762	20.1709858	Moderate	Moderate
31	7	11.5866172	20.1375458	Moderate	Moderate
32	7	11.5792546	20.1042662	Moderate	Moderate
33	7	11.5718884	20.0711460	Moderate	Moderate
34	7	11.5645186	20.0381841	Moderate	Moderate
35	7	11.5571453	20.0053796	Moderate	Moderate
36	7	11.5497686	19.9727314	Moderate	Moderate
37	7	11.5423884	19.9402385	Moderate	Moderate
38	7	11.5350049	19.9078999	Moderate	Moderate
39	7	11.5276181	19.8757145	Moderate	Moderate

40	7	11.5202280	19.8436815	Moderate	Moderate
41	7	11.5128347	19.8117998	Moderate	Moderate
42	7	11.5054382	19.7800684	Moderate	Moderate
43	7	11.4980386	19.7484865	Moderate	Moderate
44	7	11.4906359	19.7170529	Moderate	Moderate
45	7	11.4832302	19.6857669	Moderate	Moderate
46	7	11.4758215	19.6546274	Moderate	Moderate
47	7	11.4684098	19.6236335	Moderate	Moderate
48	7	11.4609953	19.5927842	Moderate	Moderate
0	8	11.9823028	22.1390575	Moderate	Moderate
1	8	11.9746247	22.0961816	Moderate	Moderate
2	8	11.9669424	22.0535340	Moderate	Moderate
3	8	11.9592560	22.0111129	Moderate	Moderate
4	8	11.9515655	21.9689166	Moderate	Moderate
5	8	11.9438709	21.9269435	Moderate	Moderate
6	8	11.9361724	21.8851920	Moderate	Moderate
7	8	11.9284699	21.8436604	Moderate	Moderate
8	8	11.9207634	21.8023470	Moderate	Moderate
9	8	11.9130531	21.7612504	Moderate	Moderate
10	8	11.9053390	21.7203690	Moderate	Moderate
11	8	11.8976210	21.6797011	Moderate	Moderate
12	8	11.8898993	21.6392452	Moderate	Moderate
13	8	11.8821739	21.5989998	Moderate	Moderate
14	8	11.8744449	21.5589634	Moderate	Moderate
15	8	11.8667123	21.5191344	Moderate	Moderate
16	8	11.8589760	21.4795114	Moderate	Moderate
17	8	11.8512363	21.4400929	Moderate	Moderate
18	8	11.8434931	21.4008774	Moderate	Moderate
19	8	11.8357465	21.3618635	Moderate	Moderate
20	8	11.8279964	21.3230498	Moderate	Moderate
21	8	11.8202431	21.2844347	Moderate	Moderate
22	8	11.8124864	21.2460170	Moderate	Moderate
23	8	11.8047266	21.2077952	Moderate	Moderate
24	8	11.7969635	21.1697680	Moderate	Moderate
25	8	11.7891972	21.1319339	Moderate	Moderate
26	8	11.7814279	21.0942916	Moderate	Moderate
27	8	11.7736555	21.0568397	Moderate	Moderate
28	8	11.7658801	21.0195770	Moderate	Moderate
29	8	11.7581018	20.9825021	Moderate	Moderate
30	8	11.7503205	20.9456137	Moderate	Moderate
31	8	11.7425364	20.9089105	Moderate	Moderate
32	8	11.7347494	20.8723912	Moderate	Moderate
33	8	11.7269597	20.8360546	Moderate	Moderate
34	8	11.7191673	20.7998993	Moderate	Moderate
35	8	11.7113722	20.7639242	Moderate	Moderate
36	8	11.7035745	20.7281279	Moderate	Moderate
37	8	11.6957742	20.6925093	Moderate	Moderate
38	8	11.6879714	20.6570672	Moderate	Moderate
39	8	11.6801662	20.6218002	Moderate	Moderate
40	8	11.6723585	20.5867073	Moderate	Moderate
41	8	11.6645485	20.5517873	Moderate	Moderate

42	8	11.6567361	20.5170389	Moderate	Moderate
43	8	11.6489215	20.4824610	Moderate	Moderate
44	8	11.6411046	20.4480524	Moderate	Moderate
45	8	11.6332856	20.4138121	Moderate	Moderate
46	8	11.6254645	20.3797388	Moderate	Moderate
47	8	11.6176413	20.3458315	Moderate	Moderate
48	8	11.6098161	20.3120890	Moderate	Moderate
0	9	12.1464820	23.1367272	Moderate	Moderate
1	9	12.1383436	23.0889782	Moderate	Moderate
2	9	12.1302020	23.0414972	Moderate	Moderate
3	9	12.1220572	22.9942821	Moderate	Moderate
4	9	12.1139092	22.9473307	Moderate	Moderate
5	9	12.1057580	22.9006410	Moderate	Moderate
6	9	12.0976038	22.8542109	Moderate	Moderate
7	9	12.0894466	22.8080384	Moderate	Moderate
8	9	12.0812863	22.7621214	Moderate	Moderate
9	9	12.0731232	22.7164580	Moderate	Moderate
10	9	12.0649571	22.6710461	Moderate	Moderate
11	9	12.0567883	22.6258839	Moderate	Moderate
12	9	12.0486166	22.5809694	Moderate	Moderate
13	9	12.0404421	22.5363006	Moderate	Moderate
14	9	12.0322650	22.4918757	Moderate	Moderate
15	9	12.0240852	22.4476928	Moderate	Moderate
16	9	12.0159029	22.4037500	Moderate	Moderate
17	9	12.0077179	22.3600456	Moderate	Moderate
18	9	11.9995305	22.3165776	Moderate	Moderate
19	9	11.9913406	22.2733442	Moderate	Moderate
20	9	11.9831483	22.2303438	Moderate	Moderate
21	9	11.9749536	22.1875745	Moderate	Moderate
22	9	11.9667567	22.1450346	Moderate	Moderate
23	9	11.9585574	22.1027223	Moderate	Moderate
24	9	11.9503560	22.0606360	Moderate	Moderate
25	9	11.9421524	22.0187739	Moderate	Moderate
26	9	11.9339467	21.9771343	Moderate	Moderate
27	9	11.9257390	21.9357156	Moderate	Moderate
28	9	11.9175292	21.8945162	Moderate	Moderate
29	9	11.9093175	21.8535344	Moderate	Moderate
30	9	11.9011039	21.8127686	Moderate	Moderate
31	9	11.8928884	21.7722172	Moderate	Moderate
32	9	11.8846712	21.7318786	Moderate	Moderate
33	9	11.8764521	21.6917512	Moderate	Moderate
34	9	11.8682314	21.6518336	Moderate	Moderate
35	9	11.8600090	21.6121241	Moderate	Moderate
36	9	11.8517851	21.5726212	Moderate	Moderate
37	9	11.8435596	21.5333235	Moderate	Moderate
38	9	11.8353325	21.4942293	Moderate	Moderate
39	9	11.8271041	21.4553374	Moderate	Moderate
40	9	11.8188743	21.4166460	Moderate	Moderate
41	9	11.8106431	21.3781540	Moderate	Moderate
42	9	11.8024107	21.3398597	Moderate	Moderate
43	9	11.7941770	21.3017617	Moderate	Moderate

44	9	11.7859421	21.2638587	Moderate	Moderate
45	9	11.7777062	21.2261492	Moderate	Moderate
46	9	11.7694691	21.1886319	Moderate	Moderate
47	9	11.7612311	21.1513054	Moderate	Moderate
48	9	11.7529921	21.1141683	Moderate	Moderate
0	10	12.3046931	24.2677510	Moderate	Moderate
1	10	12.2960862	24.2141286	Moderate	Moderate
2	10	12.2874772	24.1608250	Moderate	Moderate
3	10	12.2788661	24.1078374	Moderate	Moderate
4	10	12.2702529	24.0551630	Moderate	Moderate
5	10	12.2616377	24.0027993	Moderate	Moderate
6	10	12.2530206	23.9507436	Moderate	Moderate
7	10	12.2444016	23.8989933	Moderate	Moderate
8	10	12.2357808	23.8475459	Moderate	Moderate
9	10	12.2271582	23.7963987	Moderate	Moderate
10	10	12.2185338	23.7455493	Moderate	Moderate
11	10	12.2099077	23.6949953	Moderate	Moderate
12	10	12.2012800	23.6447340	Moderate	Moderate
13	10	12.1926507	23.5947631	Moderate	Moderate
14	10	12.1840199	23.5450803	Moderate	Moderate
15	10	12.1753875	23.4956831	Moderate	Moderate
16	10	12.1667537	23.4465691	Moderate	Moderate
17	10	12.1581186	23.3977361	Moderate	Moderate
18	10	12.1494820	23.3491818	Moderate	Moderate
19	10	12.1408442	23.3009038	Moderate	Moderate
20	10	12.1322052	23.2528999	Moderate	Moderate
21	10	12.1235650	23.2051680	Moderate	Moderate
22	10	12.1149236	23.1577057	Moderate	Moderate
23	10	12.1062811	23.1105110	Moderate	Moderate
24	10	12.0976377	23.0635816	Moderate	Moderate
25	10	12.0889932	23.0169155	Moderate	Moderate
26	10	12.0803478	22.9705105	Moderate	Moderate
27	10	12.0717016	22.9243645	Moderate	Moderate
28	10	12.0630545	22.8784756	Moderate	Moderate
29	10	12.0544066	22.8328415	Moderate	Moderate
30	10	12.0457581	22.7874604	Moderate	Moderate
31	10	12.0371089	22.7423302	Moderate	Moderate
32	10	12.0284590	22.6974489	Moderate	Moderate
33	10	12.0198086	22.6528146	Moderate	Moderate
34	10	12.0111577	22.6084253	Moderate	Moderate
35	10	12.0025064	22.5642792	Moderate	Moderate
36	10	11.9938546	22.5203743	Moderate	Moderate
37	10	11.9852025	22.4767087	Moderate	Moderate
38	10	11.9765501	22.4332807	Moderate	Moderate
39	10	11.9678975	22.3900882	Moderate	Moderate
40	10	11.9592447	22.3471296	Moderate	Moderate
41	10	11.9505917	22.3044030	Moderate	Moderate
42	10	11.9419387	22.2619067	Moderate	Moderate
43	10	11.9332857	22.2196388	Moderate	Moderate
44	10	11.9246327	22.1775976	Moderate	Moderate
45	10	11.9159797	22.1357814	Moderate	Moderate

46	10	11.9073270	22.0941885	Moderate	Moderate
47	10	11.8986744	22.0528172	Moderate	Moderate
48	10	11.8900220	22.0116658	Moderate	Moderate
0	11	12.4564012	25.5577839	Moderate	Severe
1	11	12.4473215	25.4969952	Moderate	Severe
2	11	12.4382410	25.4365912	Moderate	Severe
3	11	12.4291597	25.3765683	Moderate	Severe
4	11	12.4200777	25.3169230	Moderate	Severe
5	11	12.4109950	25.2576517	Moderate	Severe
6	11	12.4019118	25.1987512	Moderate	Severe
7	11	12.3928280	25.1402180	Moderate	Severe
8	11	12.3837436	25.0820488	Moderate	Severe
9	11	12.3746588	25.0242403	Moderate	Severe
10	11	12.3655737	24.9667892	Moderate	Moderate
11	11	12.3564881	24.9096924	Moderate	Moderate
12	11	12.3474023	24.8529466	Moderate	Moderate
13	11	12.3383162	24.7965487	Moderate	Moderate
14	11	12.3292299	24.7404955	Moderate	Moderate
15	11	12.3201434	24.6847842	Moderate	Moderate
16	11	12.3110569	24.6294115	Moderate	Moderate
17	11	12.3019703	24.5743746	Moderate	Moderate
18	11	12.2928837	24.5196704	Moderate	Moderate
19	11	12.2837972	24.4652961	Moderate	Moderate
20	11	12.2747108	24.4112487	Moderate	Moderate
21	11	12.2656245	24.3575254	Moderate	Moderate
22	11	12.2565385	24.3041235	Moderate	Moderate
23	11	12.2474528	24.2510400	Moderate	Moderate
24	11	12.2383673	24.1982723	Moderate	Moderate
25	11	12.2292823	24.1458176	Moderate	Moderate
26	11	12.2201976	24.0936732	Moderate	Moderate
27	11	12.2111135	24.0418365	Moderate	Moderate
28	11	12.2020299	23.9903049	Moderate	Moderate
29	11	12.1929469	23.9390757	Moderate	Moderate
30	11	12.1838645	23.8881464	Moderate	Moderate
31	11	12.1747828	23.8375144	Moderate	Moderate
32	11	12.1657018	23.7871772	Moderate	Moderate
33	11	12.1566217	23.7371324	Moderate	Moderate
34	11	12.1475424	23.6873775	Moderate	Moderate
35	11	12.1384640	23.6379100	Moderate	Moderate
36	11	12.1293866	23.5887276	Moderate	Moderate
37	11	12.1203102	23.5398278	Moderate	Moderate
38	11	12.1112349	23.4912084	Moderate	Moderate
39	11	12.1021607	23.4428670	Moderate	Moderate
40	11	12.0930876	23.3948013	Moderate	Moderate
41	11	12.0840158	23.3470091	Moderate	Moderate
42	11	12.0749453	23.2994880	Moderate	Moderate
43	11	12.0658761	23.2522359	Moderate	Moderate
44	11	12.0568083	23.2052506	Moderate	Moderate
45	11	12.0477420	23.1585299	Moderate	Moderate
46	11	12.0386771	23.1120716	Moderate	Moderate
47	11	12.0296138	23.0658736	Moderate	Moderate

48	11	12.0205522	23.0199339	Moderate	Moderate
0	12	12.6012251	27.0398580	Moderate	Severe
1	12	12.5916723	26.9702017	Moderate	Severe
2	12	12.5821201	26.9010176	Moderate	Severe
3	12	12.5725688	26.8323010	Moderate	Severe
4	12	12.5630182	26.7640472	Moderate	Severe
5	12	12.5534685	26.6962516	Moderate	Severe
6	12	12.5439196	26.6289096	Moderate	Severe
7	12	12.5343718	26.5620168	Moderate	Severe
8	12	12.5248249	26.4955688	Moderate	Severe
9	12	12.5152791	26.4295611	Moderate	Severe
10	12	12.5057344	26.3639896	Moderate	Severe
11	12	12.4961909	26.2988498	Moderate	Severe
12	12	12.4866486	26.2341376	Moderate	Severe
13	12	12.4771075	26.1698490	Moderate	Severe
14	12	12.4675678	26.1059797	Moderate	Severe
15	12	12.4580295	26.0425258	Moderate	Severe
16	12	12.4484925	25.9794833	Moderate	Severe
17	12	12.4389570	25.9168482	Moderate	Severe
18	12	12.4294231	25.8546168	Moderate	Severe
19	12	12.4198907	25.7927850	Moderate	Severe
20	12	12.4103600	25.7313493	Moderate	Severe
21	12	12.4008309	25.6703058	Moderate	Severe
22	12	12.3913036	25.6096508	Moderate	Severe
23	12	12.3817780	25.5493808	Moderate	Severe
24	12	12.3722543	25.4894921	Moderate	Severe
25	12	12.3627325	25.4299812	Moderate	Severe
26	12	12.3532126	25.3708446	Moderate	Severe
27	12	12.3436947	25.3120788	Moderate	Severe
28	12	12.3341789	25.2536804	Moderate	Severe
29	12	12.3246651	25.1956461	Moderate	Severe
30	12	12.3151535	25.1379725	Moderate	Severe
31	12	12.3056441	25.0806563	Moderate	Severe
32	12	12.2961369	25.0236943	Moderate	Severe
33	12	12.2866321	24.9670832	Moderate	Moderate
34	12	12.2771296	24.9108200	Moderate	Moderate
35	12	12.2676296	24.8549014	Moderate	Moderate
36	12	12.2581320	24.7993243	Moderate	Moderate
37	12	12.2486369	24.7440858	Moderate	Moderate
38	12	12.2391444	24.6891828	Moderate	Moderate
39	12	12.2296545	24.6346122	Moderate	Moderate
40	12	12.2201672	24.5803712	Moderate	Moderate
41	12	12.2106828	24.5264569	Moderate	Moderate
42	12	12.2012010	24.4728663	Moderate	Moderate
43	12	12.1917222	24.4195966	Moderate	Moderate
44	12	12.1822462	24.3666451	Moderate	Moderate
45	12	12.1727731	24.3140088	Moderate	Moderate
46	12	12.1633030	24.2616852	Moderate	Moderate
47	12	12.1538360	24.2096713	Moderate	Moderate
48	12	12.1443721	24.1579647	Moderate	Moderate
0	13	12.7389328	28.7575639	Moderate	Severe

1	13	12.7289102	28.6767524	Moderate	Severe
2	13	12.7188900	28.5965325	Moderate	Severe
3	13	12.7088722	28.5168976	Moderate	Severe
4	13	12.6988569	28.4378413	Moderate	Severe
5	13	12.6888441	28.3593574	Moderate	Severe
6	13	12.6788339	28.2814395	Moderate	Severe
7	13	12.6688264	28.2040817	Moderate	Severe
8	13	12.6588215	28.1272778	Moderate	Severe
9	13	12.6488194	28.0510220	Moderate	Severe
10	13	12.6388200	27.9753083	Moderate	Severe
11	13	12.6288235	27.9001311	Moderate	Severe
12	13	12.6188299	27.8254846	Moderate	Severe
13	13	12.6088392	27.7513633	Moderate	Severe
14	13	12.5988514	27.6777616	Moderate	Severe
15	13	12.5888668	27.6046740	Moderate	Severe
16	13	12.5788852	27.5320953	Moderate	Severe
17	13	12.5689067	27.4600202	Moderate	Severe
18	13	12.5589314	27.3884433	Moderate	Severe
19	13	12.5489594	27.3173597	Moderate	Severe
20	13	12.5389906	27.2467642	Moderate	Severe
21	13	12.5290252	27.1766518	Moderate	Severe
22	13	12.5190632	27.1070176	Moderate	Severe
23	13	12.5091046	27.0378567	Moderate	Severe
24	13	12.4991495	26.9691644	Moderate	Severe
25	13	12.4891979	26.9009359	Moderate	Severe
26	13	12.4792499	26.8331666	Moderate	Severe
27	13	12.4693056	26.7658518	Moderate	Severe
28	13	12.4593649	26.6989871	Moderate	Severe
29	13	12.4494280	26.6325679	Moderate	Severe
30	13	12.4394949	26.5665899	Moderate	Severe
31	13	12.4295656	26.5010487	Moderate	Severe
32	13	12.4196402	26.4359400	Moderate	Severe
33	13	12.4097187	26.3712595	Moderate	Severe
34	13	12.3998012	26.3070031	Moderate	Severe
35	13	12.3898878	26.2431666	Moderate	Severe
36	13	12.3799784	26.1797460	Moderate	Severe
37	13	12.3700732	26.1167372	Moderate	Severe
38	13	12.3601722	26.0541364	Moderate	Severe
39	13	12.3502754	25.9919395	Moderate	Severe
40	13	12.3403829	25.9301426	Moderate	Severe
41	13	12.3304947	25.8687421	Moderate	Severe
42	13	12.3206109	25.8077341	Moderate	Severe
43	13	12.3107316	25.7471149	Moderate	Severe
44	13	12.3008568	25.6868808	Moderate	Severe
45	13	12.2909865	25.6270282	Moderate	Severe
46	13	12.2811207	25.5675535	Moderate	Severe
47	13	12.2712597	25.5084532	Moderate	Severe
48	13	12.2614033	25.4497238	Moderate	Severe
0	14	12.8694284	30.7700461	Moderate	Severe
1	14	12.8589427	30.6749226	Moderate	Severe
2	14	12.8484612	30.5805579	Moderate	Severe

3	14	12.8379839	30.4869428	Moderate	Severe
4	14	12.8275110	30.3940684	Moderate	Severe
5	14	12.8170424	30.3019256	Moderate	Severe
6	14	12.8065782	30.2105058	Moderate	Severe
7	14	12.7961185	30.1198004	Moderate	Severe
8	14	12.7856633	30.0298009	Moderate	Severe
9	14	12.7752126	29.9404989	Moderate	Severe
10	14	12.7647665	29.8518864	Moderate	Severe
11	14	12.7543251	29.7639552	Moderate	Severe
12	14	12.7438883	29.6766974	Moderate	Severe
13	14	12.7334563	29.5901052	Moderate	Severe
14	14	12.7230290	29.5041710	Moderate	Severe
15	14	12.7126066	29.4188871	Moderate	Severe
16	14	12.7021890	29.3342461	Moderate	Severe
17	14	12.6917764	29.2502408	Moderate	Severe
18	14	12.6813687	29.1668638	Moderate	Severe
19	14	12.6709661	29.0841082	Moderate	Severe
20	14	12.6605685	29.0019669	Moderate	Severe
21	14	12.6501761	28.9204330	Moderate	Severe
22	14	12.6397887	28.8394998	Moderate	Severe
23	14	12.6294066	28.7591605	Moderate	Severe
24	14	12.6190298	28.6794087	Moderate	Severe
25	14	12.6086582	28.6002379	Moderate	Severe
26	14	12.5982920	28.5216416	Moderate	Severe
27	14	12.5879312	28.4436136	Moderate	Severe
28	14	12.5775758	28.3661477	Moderate	Severe
29	14	12.5672259	28.2892378	Moderate	Severe
30	14	12.5568815	28.2128779	Moderate	Severe
31	14	12.5465428	28.1370622	Moderate	Severe
32	14	12.5362096	28.0617847	Moderate	Severe
33	14	12.5258821	27.9870397	Moderate	Severe
34	14	12.5155603	27.9128215	Moderate	Severe
35	14	12.5052443	27.8391247	Moderate	Severe
36	14	12.4949341	27.7659436	Moderate	Severe
37	14	12.4846298	27.6932728	Moderate	Severe
38	14	12.4743313	27.6211071	Moderate	Severe
39	14	12.4640388	27.5494411	Moderate	Severe
40	14	12.4537523	27.4782697	Moderate	Severe
41	14	12.4434719	27.4075877	Moderate	Severe
42	14	12.4331975	27.3373901	Moderate	Severe
43	14	12.4229293	27.2676719	Moderate	Severe
44	14	12.4126672	27.1984282	Moderate	Severe
45	14	12.4024113	27.1296541	Moderate	Severe
46	14	12.3921618	27.0613449	Moderate	Severe
47	14	12.3819185	26.9934959	Moderate	Severe
48	14	12.3716816	26.9261024	Moderate	Severe
0	15	12.9927343	33.1601684	Moderate	Severe
1	15	12.9817950	33.0462332	Moderate	Severe
2	15	12.9708618	32.9333000	Moderate	Severe
3	15	12.9599349	32.8213552	Moderate	Severe
4	15	12.9490142	32.7103856	Moderate	Severe

5	15	12.9380998	32.6003782	Moderate	Severe
6	15	12.9271917	32.4913200	Moderate	Severe
7	15	12.9162900	32.3831987	Moderate	Severe
8	15	12.9053947	32.2760018	Moderate	Severe
9	15	12.8945059	32.1697171	Moderate	Severe
10	15	12.8836236	32.0643330	Moderate	Severe
11	15	12.8727479	31.9598375	Moderate	Severe
12	15	12.8618787	31.8562193	Moderate	Severe
13	15	12.8510162	31.7534671	Moderate	Severe
14	15	12.8401604	31.6515698	Moderate	Severe
15	15	12.8293113	31.5505166	Moderate	Severe
16	15	12.8184689	31.4502967	Moderate	Severe
17	15	12.8076334	31.3508997	Moderate	Severe
18	15	12.7968047	31.2523152	Moderate	Severe
19	15	12.7859829	31.1545331	Moderate	Severe
20	15	12.7751680	31.0575434	Moderate	Severe
21	15	12.7643601	30.9613364	Moderate	Severe
22	15	12.7535593	30.8659023	Moderate	Severe
23	15	12.7427655	30.7712317	Moderate	Severe
24	15	12.7319788	30.6773154	Moderate	Severe
25	15	12.7211992	30.5841440	Moderate	Severe
26	15	12.7104269	30.4917087	Moderate	Severe
27	15	12.6996617	30.4000006	Moderate	Severe
28	15	12.6889039	30.3090110	Moderate	Severe
29	15	12.6781533	30.2187313	Moderate	Severe
30	15	12.6674101	30.1291530	Moderate	Severe
31	15	12.6566743	30.0402679	Moderate	Severe
32	15	12.6459460	29.9520678	Moderate	Severe
33	15	12.6352251	29.8645448	Moderate	Severe
34	15	12.6245118	29.7776908	Moderate	Severe
35	15	12.6138060	29.6914982	Moderate	Severe
36	15	12.6031078	29.6059592	Moderate	Severe
37	15	12.5924172	29.5210664	Moderate	Severe
38	15	12.5817344	29.4368123	Moderate	Severe
39	15	12.5710593	29.3531897	Moderate	Severe
40	15	12.5603919	29.2701913	Moderate	Severe
41	15	12.5497323	29.1878102	Moderate	Severe
42	15	12.5390806	29.1060392	Moderate	Severe
43	15	12.5284368	29.0248717	Moderate	Severe
44	15	12.5178009	28.9443008	Moderate	Severe
45	15	12.5071730	28.8643198	Moderate	Severe
46	15	12.4965531	28.7849223	Moderate	Severe
47	15	12.4859413	28.7061018	Moderate	Severe
48	15	12.4753375	28.6278519	Moderate	Severe

APPENDIX D. HIERARCHICAL TASK ANALYSIS OF STATIONS

A. BRIDGE WATCH STATIONS

1. Navigator of the Watch (E-5)

a. Update Position on Electronic Chart Display Information System (ECDIS) (Uses TRANSIS? Electronic Charts).
   - GPS Interface updates automatically, computes PIM information as well as set drift and wind figures.

b. Change Track on ECDIS (initial track planned out in port before leaving)
   - Use track ball to move cursor over “Route” tab
   - Click “Route” tab
   - Track ball cursor over position of desired waypoint
   - Click over position of desired waypoint
   - Repeat steps 3 and 4 for as many waypoints as desired.
   - Track ball over to speed/time information for waypoints
   - Enter desired speed or time to be at that waypoint.
   - Computer calculates necessary time or arrival or SOA.

c. Designate tracks on Automatic Radar Plotting Agent (ARPA) (secondary task)
   - Track ball cursor over contact of interest.
   - Press designate button
   - Computer will automatically compute contact’s course and speed over several RADAR sweeps.

d. Add RADAR contacts to ECDIS.
   - Completed automatically once contact is designated.

e. Update Deck Log (hardest task)
   - When an order is given, write order in log book.
   - Log commanding officer’s arrival and departure from bridge

- Log all special events such as flight quarters or anomalies.

f. Update Position Log (hardest task) [possibly implement voice recorder??]
    - Every half hour note ships position from GPS displays (WRN-6 or Kelvin Hughes)
    - Record position in designated blanks corresponding to current time.
    - Note ship's course and speed from indicators in front of watch station
    - Record ship's course and speed in designated blanks corresponding to current time.
    - Note current wind and barometer readout
    - Record current information in designated blanks corresponding to current time.

g. Tertiary tasks:
    - Assist in Answering phone.
    - Assist OOD as required.

2. Officer of the Deck/ Conning Officer/ Helm/ Lee Helm (E7–O2)

a. Designate tracks on ARPA (primary)
    - Track ball cursor over contact of interest.
    - Press designate button
    - Computer will automatically compute contact's course and speed over several RADAR sweeps.

b. Change course using autopilot.
    - Press green or red button to alter course port or stbd. respectively (One push changes 1 degree, press and hold for greater course change).
    - Autopilot controls nozzles to accommodate course change and steadies on new course.

c. Change autopilot rate of turn
    - Turn ROT knob until it aligns with desired rate of turn (deg/min)

d. Adjust weather control
    - Turn “Weather Control” knob until desired setting is reached (fine – advanced)

e. Adjust input compass source
    - Push large blue button for magnetic or gyro-compass.

f. Change course manually
    - Turn wheel on arm of OOD chair left or right to change rate of turn in that direction. More wheel turn = larger rate of turn.
    - Note gyrocompass repeater to keep track of ships heading
    - When head comes close to desired heading, turn wheel in opposite direction momentarily to check swing of ship.
    - Make minor adjustments to maintain new desired heading.

g. Engage/ disengage autopilot
    - Toggle switch into either “auto” or “manual” mode, whichever is desired.

h. Turn using a combination of autopilot and manual steering
    - Disengage Autopilot by toggling switch to “manual”
    - Turn wheel to adjust desired rate of turn
    - Note gyrocompass repeater to keep track of ships head.
    - When ships head comes close o desired heading, turn wheel in opposite direction to check swing of ship.
    - Adjust new desired course in autopilot (2.b)
    - Engage Autopilot by toggling switch to “auto”

i. Adjust Throttle
    - Push throttle levers up to increase speed or down to decrease speed.
    - Note resulting RPM on engine telegraph
    - Readjust throttles as necessary to show desired RPM/ bucket settings

3. Engineering Officer of the Watch (EOOW)

a. Engineering plant monitoring
   - Ball tab pointer on EOOW computer screen to “main” screen
   - Ball tab pointer over desired feature (status, temps., hydraulics, etc)
   - Click ball tab and review screen using pointer/ ball tab.

b. Start Main Engines
   - Dispatch rover via handheld radio to desired engine room (if time permits) [good area for modeling interruptions]
   - Have Rover bar-over engine (manually turnover to move oil to main bearings, etc) (if time permits)
   - Have Rover blow down engine (remove leftover combustion gasses from cylinders) (if time permits)
   - Press “pre-lube” button to start hydraulics
   - Press “Start” button to start engine
   - Have Rover detach trailing pump (if engine was offline, shafting still turns due to impeller being dragged along in the water)
   - Press “clutch-in” button to engage engine on shaft
   - Engines have automated monitoring systems to step down speed or shut off engine in the event of an engineering casualty

c. Stop Main Engines
   - Press “Stop” button
   - Engine stops and automatically clutches out
   - Send Rover to attach trailing pump

d. Monitor/Change Generator Status
   - Ball tab on generator computer to desired generator
   - Click on generator button (1 of 4)
   - Ball tab over desired function and click (start/ stop/ change priority)

e. Generators automatically come online in order of priority when load reaches 80% of current capacity.

f. Generators automatically drop offline in reverse order of priority when load drops below 60% of current capacity.

g. T-Foil deployment
    - Press “select” button on t-foil control menu to get “deploy” option
    - Toggle selector select switch to “backup”
    - Toggle control r switch to “unlock” position to unlock t-foil from storage position
    - Toggle selector select switch back to “auto”
    - Press “deploy” Button on t-foil control menu to deploy t-foil
    - Computer automatically controls t-foil planes

h. T-Foil retract
    - Press “select” button on t-foil control menu to get “retract” option
    - Press “retract” button on t-foil control menu to retract t-foil
    - Toggle selector switch to “backup” position
    - Toggle control switch to “lock” position to lock t-foil in place
    - Toggle selector switch to “auto” position.

i. Trim Tab control
    - Press “select” button on t-foil/trim tab control panel to display trim tab control buttons
    - Press “motion on” tab
    - Computer controls trim tabs to assist in compensating ride of ship.
    - Press “select” button on t-foil/trim tab control panel to display trim tab control buttons
    - Press “motion off” tab to disable trim tab operations

j. Activate CO2 system.
    - Press “System Activate” button. Alarm will sound in space.
    - Press “Machinery Shutdown” button

- When personnel have evacuated space and all vent dampers are shut, press “open valve” button.
- Press “Release Gas” button to dump CO2 into space. (total time from 1 to 4 approx 20 seconds)

k. Establish communications
   - Plug in headset to control panel and donn headset
   - Push desired engineering space button
   - Talk into voice activated microphone.

l. The following systems are all one button push control with dedicated buttons
   - Ballast pump
   - Vent Fans
   - Hydraulic Pumps
   - Helo Foam activation
   - Camera selection
   - Fire pumps
   - Sprinkler pumps (align valves first)
   - Magnetic Fire Door operation

B. ENGINEERING ROVER ROUND

1. Engineering Rover

a. T-Foil Void
   - Enter T-Foil void area and descend ladder.
   - Inspect hydraulic pumps for normal operation/ leaks
   - Inspect hydraulic manifolds for leaks
   - Inspect bilge for water/ fluid buildup.
   - Exit Space

b. Transit to Void 3 (water void)
c. Void 3
   - Climb down hatch and two ladders to Void 3
   - Inspect Fire Pump
   - Inspect Sprinkler Pump
   - Inspect Fresh Water Pump
   - Inspect all hoses and bilge for leaks

- Exit Space

d. Transit to Port Jet Room
e. Port Jet Room
    - Climb down hatch and two ladders to Jet Room
    - Inspect hydraulic pumps
    - Inspect hydraulic hoses
    - Check for water leaks around shaft seals
    - Inspect bilge for water/ fluid buildup
    - Exit Space

f. Transit to Starboard Jet Room
g. Starboard Jet Room
    - Climb down hatch and two ladders to Jet Room
    - Inspect hydraulic pumps
    - Inspect hydraulic hoses
    - Check for water leaks around shaft seals
    - Inspect bilge for water/ fluid buildup
    - Exit Space

h. Transit to Starboard Engine Room
i. Starboard Engine Room
    - Open door and go down 1 ladder to enter Engine Room
    - Inspect first main engine for irregularities
    - Inspect second engine for irregularities
    - Climb down ladder and inspect first generator
    - Climb up two ladders and inspect second generator
    - Climb down one ladder and inspect bilge.
    - Exit Space

j. Transit to Port Engine Room
k. Port Engine Room
    - Open door and go down 1 ladder to enter Engine Room
    - Inspect first main engine for irregularities
    - Inspect second engine for irregularities
    - Climb down ladder and inspect first generator
    - Climb up two ladders and inspect second generator
    - Climb down one ladder and inspect bilge.
    - Exit Space

l. Transit to Port Fuel Header level inspection port
m. Port Fuel Header
    - Unlock and open header inspection door
    - Inspect fuel level in header
    - Close and lock header inspection door

n. Transit to Port Fuel Header level inspection port
o. Port Fuel Header
    - Unlock and open header inspection door
    - Inspect fuel level in header
    - Close and lock header inspection door

p. Round Complete

APPENDIX E. JAVA CODE FOR SIMKIT MODEL

A. INTERRUPTION GENERATOR

```
package Thesis;

import simkit.*;
import simkit.random.*;
import simkit.util.*;
/**
 * Class for modelling MITIs in tasks.  HSV2 classes listen to IG
to trigger MITIs in
 * each watchstation.
 */
public class InterruptionGenerator extends SimEntityBase{
    //parameters
    /**
     *  Random  generator  for  time  between  interruptions.   How
often do they occur?
     */
    private RandomVariate timeBetweenInterruptions;
    /**
     * Random generator for the length of each interrupion.  How
large each
     * interruption is.
     */
    private RandomVariate interruptionLength;

    //state variables
    /**
     * Keeps track of total number of interruptions generated.
     */
    protected int totalNumberOfInterruptions;
    /**
     * Keeps track of the length of each interruption.
     */
    protected double lengthOfInterruption;
    /**
     *  Keeps  track  of  wether  or  not  there  is  an  interruption
event occuring
     */
    protected boolean validInterrupt;

    //Event graph

    /**
     *zero parameter constructor
     */
    public InterruptionGenerator(){
    }

    /**
     * Creates a new instance of InterruptionGenerator
     * @param tBI time between interrutions
```

```
     * @param iL lenth of interruptions
     */
    public InterruptionGenerator(RandomVariate tBI, RandomVariate
iL) {
        setTimeBetweenInterruptions(tBI);
        setInterruptionLength(iL);
    }

    //event graph
    /**
     * Resets State variables for each simulation run.
     */
    public void reset(){
        super.reset();
        totalNumberOfInterruptions=0;
        validInterrupt=false;
    }
    /**
     * Starts simulation.  Schedules a StartInterrupt.
     */
    public void doRun(){
        firePropertyChange("validInterrupt",validInterrupt);

firePropertyChange("totalNumberOfInterruptions",totalNumberOfInterrupti
ons);

waitDelay("StartInterrupt",timeBetweenInterruptions.generate());
    }
    /**
     * Triggers the MITI in listening classes.
     * Schedules an EndInterrupt
     */
    public void doStartInterrupt(){

firePropertyChange("totalNumberOfInterruptions",totalNumberOfInterrupti
ons,++totalNumberOfInterruptions);
        lengthOfInterruption=interruptionLength.generate();

firePropertyChange("lengthOfInterruption",lengthOfInterruption);
        if(validInterrupt){
            interrupt("EndInterrupt");
        }
        validInterrupt=true;
        firePropertyChange("validInterrupt",validInterrupt);

waitDelay("StartInterrupt",timeBetweenInterruptions.generate());
        waitDelay("EndInterrupt",lengthOfInterruption);
    }
    /**
     * Triggers the end of the MITI in listening classes.
     * Schedules a StartInterrupt.
     */
    public void doEndInterrupt(){
        validInterrupt=false;
        firePropertyChange("validInterrupt",validInterrupt);
    }
```

```
//setters for private parameters
    public void setTimeBetweenInterruptions(RandomVariate tBI){
        timeBetweenInterruptions=tBI;
    }
    public void setInterruptionLength(RandomVariate iL){
        interruptionLength=iL;
    }

//getters for private parameters and protected states
    public RandomVariate getTimeBetweenInterruptions(){
        return timeBetweenInterruptions;
    }
    public RandomVariate getInterruptionLength(){
        return interruptionLength;
    }
    public int getTotalNumberOfInterruptions(){
        return totalNumberOfInterruptions;
    }
    public double getLengthOfInterruption(){
      return lengthOfInterruption;
    }
    public boolean getValidinterrupt(){
        return validInterrupt;
    }
}
```

B. CONTACT GENERATOR

```
package Thesis;

import simkit.*;
import simkit.random.*;
import java.util.*;

/**
 * Generates Surface Contacts for OOD and NOOW, generates "other"
contacts
 * for TAO.  Other contacts are those that are up for engagement.
i.e., in
 * a ASUW scenerio, the other contact arrival time would be the
same as the
 * SU contacts.
 */
public class ContactGenerator extends SimEntityBase {
    // parameters
    /**
     *  Random generator for interarrival times for surface
contacts.
     */
    private RandomVariate SUInterArrivalTime;
    /**
     * Random generator for other, engageable, contacts.
     */
    private RandomVariate otherInterArrivalTime;

    //state variables
    /**
     * Keeps track of total number of SU contacts.
     */
    protected int numberOfSUContacts;
    /**
     * Keeps track of total number of other contacts.
     */
    protected int numberOfOtherContacts;

    //Event graph

    /**
     *zero parmeter construtor
     */
    public ContactGenerator(){
    }

    /**
     * Creates a new instance of SurfaceContactGenerator
     * @param SUiat surface contact interarrival times
     * @param oiat other contact interarrival times
     */
    public  ContactGenerator(RandomVariate  SUiat,  RandomVariate
oiat) {
        setSUInterArrivalTime(SUiat);
        setOtherInterArrivalTime(oiat);
```

```
        }

        /** Set initial values of all state variables */
        public void reset() {

            super.reset();

            /** StateTransitions for the Run Event */

            numberOfSUContacts = 0;
          numberOfOtherContacts = 0;
        }

        /**
         * Initiates class in simulation.
         * Schedules a New(other)Contact, NewSUcontact.
         */
        public void doRun() {

firePropertyChange("numberOfSUContacts",numberOfSUContacts);

firePropertyChange("numberOfOtherContacts",numberOfOtherContacts);

            waitDelay("NewContact",otherInterArrivalTime.generate());
            waitDelay("NewSUContact",SUInterArrivalTime.generate());
        }

        /**
         * A new other contact.  Schedules another NewContact.
         */
        public void doNewContact() {
            numberOfOtherContacts = numberOfOtherContacts+1;
            firePropertyChange("numberOfOtherContacts",
numberOfOtherContacts);


waitDelay("NewContact",otherInterArrivalTime.generate(),new
Object[]{},0);
        }

        /**
         * A new surface contact, Scedules a NewSUContact.
         */
        public void doNewSUContact() {
            numberOfSUContacts = numberOfSUContacts+1;
            firePropertyChange("numberOfSUContacts",
numberOfSUContacts);


waitDelay("NewSUContact",SUInterArrivalTime.generate(),new
Object[]{},0);
        }

        //getters for private parameters and proteted states
        public int getNumberOfSUContacts() {
            return  numberOfSUContacts;
        }
```

```
        public int getNumberOfOtherContacts() {
            return  numberOfOtherContacts;
        }
        public RandomVariate getSUInterArrivalTime() {
            return this.SUInterArrivalTime;
        }
        public RandomVariate getOtherInterArrivalTime() {
            return this.otherInterArrivalTime;
        }

        //setters for private parameters
        public        void        setSUInterArrivalTime(RandomVariate
SUInterArrivalTime) {
            this.SUInterArrivalTime = SUInterArrivalTime;
        }
        public void setOtherInterArrivalTime(RandomVariate oiat) {
            this.otherInterArrivalTime = oiat;
        }
    }
```

C. COMMAND CLASS

```
package Thesis;

import simkit.*;
import simkit.random.*;
import java.util.*;

/**
 * Class to represent the commanding officer of the ship.  Can be
used
 * to program additional tasks for the HSV model.  Also used to
model
 * strike missions.  OOD,TAO listen.
 */
public class Command extends SimEntityBase {
    //Parameters
    /**
     * Random generator for the times between command interjected
course
     * changes.
     */
    private                              simkit.random.RandomVariate
timeBetweenCommandCourseChangeGen;
    /**
     * Random generator for the times between command interjected
speed
     * changes.
     */
    private                              simkit.random.RandomVariate
timeBetweenCommandSpeedChangeGen;
    /**
     *  Random generator for the times between strike missions.
(if any)
     */
    private RandomVariate timeBetweenStrikeMissionsGen;

    //Event Graph

    /**
     *zero parameter constructor
     */
    public Command(){
    }
    /**
     * Creates a new instance of Command
     * @param tbcccg times between course changes
     * @param tbcscg times between speed changes
     * @param tbsmg times between strike missions.
     */
    public Command(simkit.random.RandomVariate tbcccg,
            simkit.random.RandomVariate   tbcscg,   RandomVariate
tbsmg) {

        setTimeBetweenCommandCourseChangeGen(tbcccg);
        setTimeBetweenCommandSpeedChangeGen(tbcscg);
```

```
            setTimeBetweenStrikeMissionsGen(tbsmg);
        }

        /** Set initial values of all state variables */
        public void reset() {

            super.reset();

            /** StateTransitions for the Run Event */

        }

        /**
         * initializes the class in the simulation.
         * Schedules CommandCourseChange, CommandSpeedChange,
         *  NewStrikeMission
         */
        public void doRun() {

waitDelay("CommandCourseChange",timeBetweenCommandCourseChangeGen.gener
ate(),new Object[]{},0);

waitDelay("CommandSpeedChange",timeBetweenCommandSpeedChangeGen.generat
e(),new Object[]{},0);

waitDelay("NewStrikeMission",timeBetweenStrikeMissionsGen.generate());
        }

        /**
         * OOD listens,
         * Shcedules a COmmandCourseChange
         */
        public void doCommandCourseChange() {
            if (true) {

waitDelay("CommandCourseChange",timeBetweenCommandCourseChangeGen.gener
ate(),new Object[]{},0);
            }
        }

        /**
         * OODListens,
         * Scedules a CommandSpeedChange
         */
        public void doCommandSpeedChange() {

waitDelay("CommandSpeedChange",timeBetweenCommandSpeedChangeGen.generat
e(),new Object[]{},0);
        }

        /**
         * TAO listens,
         * Schedules a NewStrikeMission
         */
        public void doNewStrikeMission(){

waitDelay("NewStrikeMission",timeBetweenStrikeMissionsGen.generate());
```

```
          }

          //setters and getters for private parameters
          public                                                    void
setTimeBetweenCommandCourseChangeGen(simkit.random.RandomVariate
tbcccg) {
              timeBetweenCommandCourseChangeGen = tbcccg;
          }
          public                              simkit.random.RandomVariate
getTimeBetweenCommandCourseChangeGen() {
              return timeBetweenCommandCourseChangeGen;
          }
          public                                                    void
setTimeBetweenCommandSpeedChangeGen(simkit.random.RandomVariate  tbcscg)
{
              timeBetweenCommandSpeedChangeGen = tbcscg;
          }
          public                              simkit.random.RandomVariate
getTimeBetweenCommandSpeedChangeGen() {
              return timeBetweenCommandSpeedChangeGen;
          }
          public    void    setTimeBetweenStrikeMissionsGen(RandomVariate
tbsmg){
              this.timeBetweenStrikeMissionsGen=tbsmg;
          }
          public RandomVariate getTimeBetweenStrikeMissionsGen(){
              return this.timeBetweenStrikeMissionsGen;
          }
      }
```

D. HSV2 MODEL ASSEMBLY CLASS

```
package Thesis;

import simkit.*;
import simkit.random.*;
import simkit.stat.*;
import simkit.util.*;
import java.util.Random.*;
import java.text.*;
import java.io.*;
import java.util.ArrayList;
import java.beans.PropertyChangeListener;
import java.util.*;

/**
 *
 * @author  Lorio-Diaz
 *
 *    HSV2Model incoorporates all watch classes and governing
classes with their
 * appropriate assembly to simulate watches aboard a high speed
vessel in
 * varying conditions.  Parameters may be changed as appropriate
to simulate
 * different missions.  InterruptionGenerator parameters may be
changed to
 * simulate different motion effects on the watches.  MOE is the
utilization
 * of each watch.  As utilization approaches 1, the watch stander
is not as
 * capable of completing his assigned tasks.
 *
 *  Code for ContactGenerator.    SUInterArrivalTimes are for
surface
 * contacts, listened to by NOOW and OOD.  otherInterArrivalTimes
are
 *  for TAO, "other" contacts may be engageable dependant on
mission.
 *
 * Code for InterruptionGenerator.  Listened to by all watches
 * to implement effects of MITIs.  timeBetweenInterruptions and
 * interruption length are dependent on wave direction, height,
 * and vessel speed which will vary for different missions.
 *
 * Code for EngineeringRover.  The rover is the simplest watch
and
 * listens to the interruptionGenerator for MITIs
 *
 * Code for Command.  Command class is a superior class that
 * sets intervals for mission oriented course and speed changes
as well
 * as intervals between strike missions.
 *
 * Code for NOOW.  The NOOW is directly subordinate to the OOD.
The
```

```
       * NOOW listens for SUContacts as well as course and speed
changes from
       * the OOD.
       *
       * Code for OOD.  The OOD is directly subordinate to the TAO and
       * Command classes.  Listens for Command course and speed
       * changes as well as SUContacts.  There is a probability
       * that a course or speed change will be required for each
       * new SUcontact
       *
       * Code for TAO.  The TAO is the busiest watch.  Subordinate to
the
       * Command class, the TAO listens for Strike missions and
otherContacts
       * that are engageable with a probability for each mission.
       */
      public class HSV2ModelAssembly extends BasicAssembly {

          /**
           *<p> Instantiates SimEntity and Stats objects in arrays
defined in
           * BasicASsembly superclass.  Performs hookups (must be done
here
           * because superclass constructor lacks details of assembly.
           */
          public HSV2ModelAssembly(){
          }

          /**
           *Instanciates the array of replication stat objects
           */
          public void createReplicationStats(){
              replicationStats = new SampleStatistics[]{
                  new SimpleStatsTally("lengthOfInterruption"),
                  new SimpleStatsTally("roundTime"),
                  new SimpleStatsTally("timeInterrupted"),
                  new SimpleStatsTimeVarying("erBusy"),
                  new SimpleStatsTally("timeInterrupted"),
                  new CollectionSizeTimeVaryingStats("queue"),
                  new SimpleStatsTimeVarying("noowBusy"),
                  new SimpleStatsTally("timeInterrupted"),
                  new CollectionSizeTimeVaryingStats("queue"),
                  new SimpleStatsTimeVarying("oodBusy"),
                  new SimpleStatsTally("timeInterrupted"),
                  new CollectionSizeTimeVaryingStats("queue"),
                  new SimpleStatsTimeVarying("busy")
                 };
              ((Resetter)simEntity[7]).setStats(replicationStats);
          }

          /**
           * Instanciates the array of property change listeners
           */
          public void createProperyChangeListeners(){
              propertyChangeListener = new PropertyChangeListener[] {
                  new SimplePropertyDumper()
              };
```

```
        }

        /**
         * Instanciates the array of SimEntities
         */
        protected void createSimEntities() {
            simEntity = new SimEntity[]{
                new InterruptionGenerator(),
                new Command(),
                new ContactGenerator(),
                new EngineeringRover(),
                new NavigatorOfTheWatch(),
                new OfficerOfTheDeck(),
                new TacticalActionOfficer(),
                new Resetter(20000.00)
            };
        }

        /**
         * Hook up replication stats objects here.
         * These compute the means for each run whose
         * input will be passed to the design point stats
         * objects
         */
        protected void hookupReplicationListeners() {

simEntity[0].addPropertyChangeListener(replicationStats[0]);

simEntity[3].addPropertyChangeListener(replicationStats[1]);

simEntity[3].addPropertyChangeListener(replicationStats[2]);

simEntity[3].addPropertyChangeListener(replicationStats[3]);

simEntity[4].addPropertyChangeListener(replicationStats[4]);

simEntity[4].addPropertyChangeListener(replicationStats[5]);

simEntity[4].addPropertyChangeListener(replicationStats[6]);

simEntity[5].addPropertyChangeListener(replicationStats[7]);

simEntity[5].addPropertyChangeListener(replicationStats[8]);

simEntity[5].addPropertyChangeListener(replicationStats[9]);

simEntity[6].addPropertyChangeListener(replicationStats[10]);

simEntity[6].addPropertyChangeListener(replicationStats[11]);

simEntity[6].addPropertyChangeListener(replicationStats[12]);
        }

        /**
         * All simEventListening is connected here.
         * The identities of the simEvent[i] and
         * replicationStat[j] are from the definitions
```

```
         * in the create(x) methods above.
         */
        protected void hookupSimEventListeners() {
            simEntity[0].addSimEventListener(simEntity[3]);
            simEntity[0].addSimEventListener(simEntity[4]);
            simEntity[0].addSimEventListener(simEntity[5]);
            simEntity[0].addSimEventListener(simEntity[6]);
            simEntity[1].addSimEventListener(simEntity[5]);
            simEntity[1].addSimEventListener(simEntity[6]);
            simEntity[2].addSimEventListener(simEntity[4]);
            simEntity[2].addSimEventListener(simEntity[5]);
            simEntity[2].addSimEventListener(simEntity[6]);
            simEntity[5].addSimEventListener(simEntity[4]);
            simEntity[6].addSimEventListener(simEntity[5]);
        }

        /**
         * Sets the parameters for the random variates in easch sim
entity as
         * called by the SetParameters method
         * @param index sim entity
         *  @param values hasmap of randomVaraite names and new
parameters
         */
        public void setParameter(int index, HashMap values) {
            for    (Iterator    i    =    values.keySet().iterator();
i.hasNext(); ) {
                Object key = i.next();
                Object value = values.get(key);
                simEntity[index].setProperty(key.toString(), value);
            }
        }


        /**
         * Sets the parameters for all of the simEntity objects by
calling
         * setParameter for each eimEntity.
         * @param values array of hashmaps that contain the new
         *               parameters for each simEntity
         */
        public void setParameters(HashMap[] values) {
            for (int i = 0; i < values.length; ++i) {
                setParameter(i, values[i]);
            }
        }

        /**
         * Initializes the assembly,
         * reads from input file
         * runs simulation
         * writes to output file
         * @param args the command line arguments
         * @throws Throwable prevents errors
         */
        public static void main(String[] args) throws Throwable {
            HSV2ModelAssembly HSV2 = new HSV2ModelAssembly();
```

```
            //code for input data file
            String fileName = args.length > 0 ? args[0] :
"/C:/tmp/HSVModelInput.csv";
            File file = new File(fileName);

            //code to check input file
            System.out.println("Input File: " + file);
            System.out.println("File exists? " + file.exists());

            //code for output file
            String outputFileName = args.length > 1 ? args[1] :
"/C:/tmp/output.csv";
            File outputFile = new File(outputFileName);
            /*if (outputFile.exists()) {
                System.out.println("Warning! Output file will be
overwritten - please move or delete");
                return;
            }*/
            System.out.println("Output file will be: " + outputFile);

            //reading in the data from the file
            ArrayList dataParams = new ArrayList();
            FileReader fileReader = new FileReader(file);
            BufferedReader bufferedReader = new
BufferedReader(fileReader);

            // confirm data file input correctly and add data to
arrayList dataParams
            System.out.println("Echo of data file:");
            int lineNumber = 0;
            for (String line = bufferedReader.readLine(); line !=
null; line=bufferedReader.readLine()) {
                lineNumber += 1;
                System.out.println("Line " + lineNumber + ": " +
line);
                if (lineNumber == 1) {
                    continue;
                }
                String[] rawData = line.split(",");
                dataParams.add(getData(rawData));
            }
            bufferedReader.close();

            //writing the heading for the output file
            FileWriter fileWriter = new FileWriter(outputFile);
            BufferedWriter bufferedWriter = new
BufferedWriter(fileWriter);
            bufferedWriter.write("Parameter Group,Run
#,ERUtilization,NOOWUtilization,OODUtilization,TAOUtilization");

            //run the simulation with each line from the input file

            HSV2.setSaveReplicationData(false);
            HSV2.setPrintReplicationReports(false);
            RandomVariateManager rvm = new RandomVariateManager();
            RandomVariate[] randomVariates;
```

```
            SampleStatistics[] stats;
            HashMap[] newValues = new HashMap[7];
            for (int i=0; i<newValues.length; ++i){
                newValues[i] = new HashMap();
            }
            double fixInterval;
            double numberRepititions;
            double simRunTime;
            double probabilityCourseChangeRequired;
            double probabilitySpeedChangeRequired;
            double probabilityEngageContact;
            double ERUtil;
            double OODUtil;
            double NOOWUtil;
            double TAOUtil;
            HSV2.setVerbose(false);
            HSV2.setPrintReplicationReports(false);
            HSV2.setPrintSummaryReport(false);

            System.out.println(HSV2);

            /**
             * for most missions, besides AAW, ASW, etc. in which the
             * TAO is more concerned with other engagable contacts
             */
            Adapter             adapter             =             new
Adapter("NewSUContact","NewContact");
            adapter.connect(HSV2.simEntity[2],HSV2.simEntity[6]);

            //loop for each line of the input file
            for (int i = 0; i < dataParams.size(); ++i) {

                //get data for this run
                double[]       dataForThisRun       =       (double[])
dataParams.get(i);
                System.out.println(  "Run  "  +  i  +  ":  "  +  dump(
dataForThisRun ));

                //set  simulation  parameters  using  data  from  the
arrayList dataParams

                simRunTime=dataForThisRun[0];
                numberRepititions=dataForThisRun[1];
                fixInterval=dataForThisRun[2];
                probabilityCourseChangeRequired=dataForThisRun[24];
                probabilitySpeedChangeRequired=dataForThisRun[25];
                probabilityEngageContact = dataForThisRun[45];
                rvm.setParameters(dataForThisRun);
                /*for test...
                 System.out.println(rvm);*/
                randomVariates=rvm.getVariables();


newValues[0].put("timeBetweenInterruptions",randomVariates[2]);

newValues[0].put("interruptionLength",randomVariates[3]);
```

```
newValues[1].put("timeBetweenCommandCourseChangeGen",randomVariates[7])
;

newValues[1].put("timeBetweenCommandSpeedChangeGen",randomVariates[8]);

newValues[1].put("timeBetweenStrikeMissionsGen",randomVariates[9]);

newValues[2].put("SUInterArrivalTime",randomVariates[1]);

newValues[2].put("otherInterArrivalTime",randomVariates[0]);

newValues[3].put("timeBetweenRounds",randomVariates[4]);

newValues[3].put("shortTransitTime",randomVariates[5]);

newValues[3].put("longTransitTime",randomVariates[6]);

newValues[4].put("timeToUpdateDeckLogGen",randomVariates[10]);

newValues[4].put("timeToDesignateContactGen",randomVariates[11]);

newValues[4].put("timeToUpdatePositionLogGen",randomVariates[12]);

newValues[4].put("timeToManageECTIDSGen",randomVariates[13]);

newValues[5].put("timeToDesignateContactGen",randomVariates[11]);

newValues[5].put("timeToChangeCourseGen",randomVariates[14]);

newValues[5].put("timeToCallShipGen",randomVariates[15]);

newValues[5].put("timeToChangeSpeedGen",randomVariates[16]);

newValues[5].put("timeToMonitorPositionGen",randomVariates[17]);

newValues[6].put("timeToEvaluateContactGen",randomVariates[18]);

newValues[6].put("timeToTalkOnRadioGen",randomVariates[19]);

newValues[6].put("timeToReviewEngagementGen",randomVariates[20]);

newValues[6].put("timeToMonitorGCCSGen",randomVariates[21]);

newValues[6].put("timeToMonitorPositionGen",randomVariates[17]);

newValues[6].put("timeToPlanStrikeMissionGen",randomVariates[22]);

newValues[6].put("radioIntervalGen",randomVariates[23]);

                  HSV2.simEntity[4].setProperty("fixInterval",       new
Double(fixInterval));

HSV2.simEntity[5].setProperty("probabilityCourseChangeNecessary",    new
Double(probabilityCourseChangeRequired));
```

```
HSV2.simEntity[5].setProperty("probabilitySpeedChangeNecessary",new
Double(probabilitySpeedChangeRequired));
                  HSV2.simEntity[5].setProperty("fixInterval",       new
Double(fixInterval));
                  HSV2.simEntity[6].setProperty("fixInterval",       new
Double(fixInterval));

HSV2.simEntity[6].setProperty("probabilityEngageContact",            new
Double(probabilityEngageContact));

                  HSV2.setParameters(newValues);
                  HSV2.setStopTime(dataForThisRun[0]);

                  /*System.out.println(HSV2.simEntity[0]);
                  System.out.println(HSV2.simEntity[1]);
                  System.out.println(HSV2.simEntity[2]);
                  System.out.println(HSV2.simEntity[3]);
                  System.out.println(HSV2.simEntity[4]);
                  System.out.println(HSV2.simEntity[5]);
                  System.out.println(HSV2.simEntity[6]);*/

                  //simulation management for each parameter run
                  for (int j =0; j<numberRepititions; ++j){
                      System.out.println("repititon: "+(j+1));
                      HSV2.run();
                      ERUtil=HSV2.replicationStats[3].getMean();
                      NOOWUtil=HSV2.replicationStats[6].getMean();
                      OODUtil = HSV2.replicationStats[9].getMean();
                      TAOUtil=HSV2.replicationStats[12].getMean();

                      //Collecting the results and printing results to
the output file

bufferedWriter.write(System.getProperty("line.separator"));
                      bufferedWriter.write(i + "," +(j+1)+ "," +ERUtil+
"," +NOOWUtil+
                         "," +OODUtil+ "," +TAOUtil);
                  }
                 //stats=HSV2.getDesignPointStats();
                 //
bufferedWriter.write(System.getProperty("line.separator"));
                 //System.out.println("means:"+stats[3]);
              }

              //finalize the output file
              bufferedWriter.close();
              System.out.println(outputFile + " Written");
          }

          /**
           * Takes input file and converts to array of doubles
           * @param data input data
           * @return double array from input file
           */
           public static double[] getData(String[] data) {
              double[] doubleData = new double[data.length];
```

```
        for (int i = 0; i < doubleData.length; ++i) {
            doubleData[i] = Double.parseDouble(data[i]);
        }
        return doubleData;
    }

     /**
      * Writes lines of data to mirror input
      * @param data input data
      * @return string of data from input
      */
    public static String dump(double[] data) {
        StringBuffer buf = new StringBuffer();
        for (int i = 0; i < data.length; ++i) {
            buf.append(data[i]);
            buf.append(' ');
        }
        return buf.toString();
    }
}
```

Java code for **RandomVariateManager** class for use with HSV2ModelAssembly

```
package Thesis;

import simkit.random.*;

/**
 *
 * @author  G.P. Lorio
 */
public class RandomVariateManager {

    private RandomVariate[] rvs;

    /** Creates a new instance of RandomVariateManager with the random Variates
     * required for the HSV-2 simualtion runs.  RV parameters are dummy values.
     */
    public RandomVariateManager() {
        rvs = new RandomVariate[24];
      /*Other.IA.Time*/                                    rvs[0]      = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*SU.IA.Time*/                                     rvs[1]        = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*Time.btw.Interrupt*/                               rvs[2]      = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*Interrupt.Length*/                                 rvs[3]      = RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0), new Double(.5)});
      /*Time.btwn.Rnds*/                                   rvs[4]      = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*Short.Xsit.Time*/                                  rvs[5]      = RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0), new Double(.5)});
      /*Long.Xsit.Time*/                                   rvs[6]      = RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0), new Double(.5)});
      /*Time.btwn.Crse.Change*/                            rvs[7]      = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*Time.btwn.Spd.Change*/                             rvs[8]      = RandomVariateFactory.getInstance("Exponential",new     Object[]     {new Double(1.0)});
      /*Time.btwn.Strike.Missn*/                           rvs[9]      = RandomVariateFactory.getInstance("Exponential",new       Object[]{new Double(1.0)});
      /*Time.Updte.Deck.Log*/                            rvs[10]       = RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0), new Double(.5)});
```

```
        /*Time.Dsg.Contact*/                              rvs[11]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Updte.Posit.Log*/                           rvs[12]      =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Mnge.ECTDS*/                               rvs[13]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Change.Crse*/                             rvs[14]        =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Call.Ship*/                                rvs[15]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Change.Spd*/                               rvs[16]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Mon.Posit*/                                rvs[17]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Eval.Contact*/                              rvs[18]      =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Talk.Radio*/                               rvs[19]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Review.Engage*/                             rvs[20]      =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Mon.GCCS*/                                 rvs[21]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Time.Plan.Strike*/                              rvs[22]       =
RandomVariateFactory.getInstance("Gamma",new Object[] {new Double(1.0),
new Double(.5)});
        /*Radio.Interval*/                                rvs[23]       =
RandomVariateFactory.getInstance("Exponential",new     Object[]     {new
Double(1.0)});
         }

         /**
          * Sets the paremeter values for each random variate in the
          * HSV2 model assembly
          * @param val array of double values read in from the
          * input file
          */
         public void setParameters(double[] val) {
             rvs[0].setParameters(new  Object[]  {  new  Double  (val[3])
});
             rvs[1].setParameters(new  Object[]  {  new  Double  (val[4])
});
             rvs[2].setParameters(new  Object[]  {  new  Double  (val[5])
});
             rvs[3].setParameters(new  Object[]  {  new  Double(val[6]),
new Double(val[7])});
```

```
            rvs[4].setParameters(new Object[] { new Double (val[8])
});
            rvs[5].setParameters(new Object[] { new Double(val[9]),
new Double(val[10])});
            rvs[6].setParameters(new Object[] { new Double(val[11]),
new Double(val[12])});
            rvs[7].setParameters(new Object[] { new Double (val[13])
});
            rvs[8].setParameters(new Object[] { new Double (val[14])
});
            rvs[9].setParameters(new Object[] { new Double (val[15])
});
            rvs[10].setParameters(new Object[] { new Double(val[16]),
new Double(val[17])});
            rvs[11].setParameters(new Object[] { new Double(val[18]),
new Double(val[19])});
            rvs[12].setParameters(new Object[] { new Double(val[20]),
new Double(val[21])});
            rvs[13].setParameters(new Object[] { new Double(val[22]),
new Double(val[23])});
            rvs[14].setParameters(new Object[] { new Double(val[26]),
new Double(val[27])});
            rvs[15].setParameters(new Object[] { new Double(val[28]),
new Double(val[29])});
            rvs[16].setParameters(new Object[] { new Double(val[30]),
new Double(val[31])});
            rvs[17].setParameters(new Object[] { new Double(val[32]),
new Double(val[33])});
            rvs[18].setParameters(new Object[] { new Double(val[34]),
new Double(val[35])});
            rvs[19].setParameters(new Object[] { new Double(val[36]),
new Double(val[37])});
            rvs[20].setParameters(new Object[] { new Double(val[38]),
new Double(val[39])});
            rvs[21].setParameters(new Object[] { new Double(val[40]),
new Double(val[41])});
            rvs[22].setParameters(new Object[] { new Double(val[42]),
new Double(val[43])});
            rvs[23].setParameters(new Object[] { new Double (val[44])
});
        }

        /**
         * Retrund length of random variate array
         * @return length of rvs array
         */
        public int getLength(){
            return rvs.length;
        }

        /**
         * returns the array of random variates
         * @return clone of rvs
         */
        public RandomVariate[] getVariables(){
            return ((RandomVariate[])rvs.clone());
        }
```

```
    /**
     * Lists the name position and paramters of the each rv
     * @return string of ranfom variate position and
     * parameters
     */
    public String toString() {
        StringBuffer buf = new StringBuffer();
     for (int i = 0; i < rvs.length; ++i) {
            buf.append("randomVariate[");
            buf.append(i);
            buf.append("] = ");
            buf.append(rvs[i]);
            buf.append(System.getProperty("line.separator"));
        }
        return buf.toString();
    }
}
```

E. ENGINEERING ROVER

```
package Thesis;

import simkit.*;
import simkit.random.*;
import simkit.util.*;
/**
 * The Engineering Rover class simulates the activities of the
 * Engineering Rover watch aboard the HSV-2.  The ER listens to
 * the InterruptionGenerator for MITI inputs.
 *  Methods represent the stops on a typical engineering rover
round.
 *  Times for inspections are represented as two random transit
times,
 * a short and a long.
 */
public class EngineeringRover extends SimEntityBase{
    // Parmeters
    /**
     *  Random generator for time between Engineering Rover
rounds.
     */
    private RandomVariate timeBetweenRounds;
    /**
     * Random generator for short transit times between spaces.
     */
    private RandomVariate shortTransitTime;
    /**
     * Random generator for long transit times between spaces.
     */
    private RandomVariate longTransitTime;

    // State Variables
    /**
     * Keeps track of the start time for each round
     */
    protected double roundStartTime;
    /**
     * The time that it takes to complete a round.
     */
    protected double roundTime;
    /**
     * The next task that the rover will complete.
     */
    protected String nextEvent;
    /**
     * Keeps track of the time the current task will take.
     */
    protected double timeUntilNextEvent;
    /**
     * Keeps track of the start times for each space
     */
    protected double taskStartTime;
    /**
     * The amount of task completed when an interruption occurs.
```

```
         */
        protected double workDone;
        /**
         *  Keeps track of interruptions that are in progress to
prevent other tasks
         * from starting during an interruption.
         */
        protected boolean erValidInterrupt;
        /**
         * 1 if rover is busy, 0 otherwise,
         */
        protected int erBusy;
        /**
         * Keeps track of the time that each interruption starts
         */
        protected double interruptionStartTime;
        /**
         * The length of the interruption
         */
        protected double timeInterrupted;

        //event graph

        /**
         *zero parameter constructor
         */
        public EngineeringRover(){
        }

        /**
         * Creates a new instance of EngineeringRover
         * @param tBR timeBetweenRounds
         * @param sTT shortTransitTime
         * @param lTT longTransitTime
         */
        public EngineeringRover(RandomVariate tBR, RandomVariate sTT,
RandomVariate lTT) {
            setTimeBetweenRounds(tBR);
            setShortTransitTime(sTT);
            setLongTransitTime(lTT);
        }

        //event graph
        /**
         * Resets state variables for each simulation run.
         */
        public void reset(){
            super.reset();
            taskStartTime=0.0;
            nextEvent="Start";
            timeUntilNextEvent=timeBetweenRounds.generate();
            erValidInterrupt=false;
            erBusy=0;
        }
        /**
         * Simulation Start.  Schedules StartRound.
         */
```

```
            public void doRun(){
                nextEvent="StartRound";
                timeUntilNextEvent=timeBetweenRounds.generate();
                firePropertyChange("nextEvent",nextEvent);

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                firePropertyChange("taskStartTime",taskStartTime);
                waitDelay("StartRound",timeUntilNextEvent);
            }
            /**
             * Sets round start time and sets busy =1.
             * Schedules T-FoilVoid.
             */
            public void doStartRound(){
                erBusy=1;
                firePropertyChange("erBusy",erBusy);
                roundStartTime=Schedule.getSimTime();
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                firePropertyChange("roundStartTime",roundStartTime);
                timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "TFoilVoid";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay(nextEvent,timeUntilNextEvent);

            }
            /**
             * Schedules Void3
             */
            public void doTFoilVoid(){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "Void3";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay(nextEvent,timeUntilNextEvent);
            }
            /**
             * Schedules portJetRoom
             */
            public void doVoid3(){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeUntilNextEvent = longTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "PortJetRoom";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay(nextEvent,timeUntilNextEvent);
            }
```

```
        /**
         * Schedules StarboardJetRoom
         */
        public void doPortJetRoom(){
            taskStartTime=Schedule.getSimTime();
            firePropertyChange("taskStartTime",taskStartTime);
            timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
            String lastEvent=nextEvent;
            nextEvent = "StarboardJetRoom";
            firePropertyChange("nextEvent",lastEvent,nextEvent);
            waitDelay(nextEvent,timeUntilNextEvent);
        }
        /**
         * SchedulesStarboardEngineRoom
         */
        public void doStarboardJetRoom(){
            taskStartTime=Schedule.getSimTime();
            firePropertyChange("taskStartTime",taskStartTime);
            timeUntilNextEvent = longTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
            String lastEvent=nextEvent;
            nextEvent = "StarboardEngineRoom";
            firePropertyChange("nextEvent",lastEvent,nextEvent);
            waitDelay(nextEvent,timeUntilNextEvent);
        }
        /**
         * Schedules PortEngineRoom
         */
        public void doStarboardEngineRoom(){
            taskStartTime=Schedule.getSimTime();
            firePropertyChange("taskStartTime",taskStartTime);
            timeUntilNextEvent = longTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
            String lastEvent=nextEvent;
            nextEvent = "PortEngineRoom";
            firePropertyChange("nextEvent",lastEvent,nextEvent);
            waitDelay(nextEvent,timeUntilNextEvent);
        }
        /**
         * Shcedules PortFuelHeader
         */
        public void doPortEngineRoom(){
            taskStartTime=Schedule.getSimTime();
            firePropertyChange("taskStartTime",taskStartTime);
            timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
            String lastEvent=nextEvent;
            nextEvent = "PortFuelHeader";
            firePropertyChange("nextEvent",lastEvent,nextEvent);
            waitDelay(nextEvent,timeUntilNextEvent);
        }
        /**
```

```
             * Schedules StarboardFuelHeader
             */
            public void doPortFuelHeader(){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "StarboardFuelHeader";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay(nextEvent,timeUntilNextEvent);
            }
            /**
             * SchedulesEndRound
             */
            public void doStarboardFuelHeader(){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeUntilNextEvent = shortTransitTime.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "EndRound";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay(nextEvent,timeUntilNextEvent);
            }
            /**
             * Sets Busy = 0.
             * Schedules StartRound
             */
            public void doEndRound(){
                erBusy=0;
                firePropertyChange("erBusy",erBusy);
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                roundTime=Schedule.getSimTime()-roundStartTime;
                firePropertyChange("roundTime",roundTime);
                timeUntilNextEvent = timeBetweenRounds.generate();

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                String lastEvent=nextEvent;
                nextEvent = "StartRound";
                firePropertyChange("nextEvent",lastEvent,nextEvent);
                waitDelay("StartRound",timeUntilNextEvent);
            }
            /**
             * Listens to InterruptionGenerator to cancel current task.
             */
            public void doStartInterrupt(){
                if(erBusy==1 && !erValidInterrupt){
                    interrupt(nextEvent);
                    workDone=Schedule.getSimTime()-taskStartTime;
                    erValidInterrupt=true;

firePropertyChange("erValidInterrupt",erValidInterrupt);
                    interruptionStartTime=Schedule.getSimTime();
```

```
firePropertyChange("interruptionStartTime",interruptionStartTime);
            }
        }
        /**
         * Listens to InterruptionGenerator to reschedule the current
task
         */
        public void doEndInterrupt(){
            if (erValidInterrupt){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeUntilNextEvent= timeUntilNextEvent-workDone;
                firePropertyChange("nextEvent",nextEvent);

firePropertyChange("timeUntilNextEvent",timeUntilNextEvent);
                timeInterrupted=Schedule.getSimTime()-
interruptionStartTime;

firePropertyChange("timeInterrupted",timeInterrupted);
                erValidInterrupt=false;

firePropertyChange("erValidInterrupt",erValidInterrupt);
                waitDelay(nextEvent,timeUntilNextEvent,1.0);

            }
        }
        //setters for private parameters
        public void setTimeBetweenRounds(RandomVariate tBR){
            timeBetweenRounds=tBR;
        }
            public void setShortTransitTime(RandomVariate sTT){
            shortTransitTime=sTT;
        }
        public void setLongTransitTime(RandomVariate lTT){
            longTransitTime=lTT;
        }

        //getters for private parameters and protected states
        public RandomVariate getTimeBetweenRounds(){
            return timeBetweenRounds;
        }
        public RandomVariate getShortTransitTime(){
            return shortTransitTime;
        }
        public RandomVariate getLongTransitTime(){
            return longTransitTime;
        }
        public double getRoundStartTime(){
            return roundStartTime;
        }
        public double getRoundTime(){
            return roundTime;
        }
        public double getTimeUntilNextEvent(){
            return timeUntilNextEvent;
        }
```

```
    public String getNextEvent(){
        return nextEvent;
    }
    public double getTaskStartTime(){
        return taskStartTime;
    }
    public double getWorkDone(){
        return workDone;
    }
    public double geterBusy(){
        return erBusy;
    }
    public double getInterruptionStartTime(){
        return this.interruptionStartTime;
    }
    public double getTimeinterrupted(){
        return this.timeInterrupted;
    }
    public boolean getErValidInterrupt(){
        return this.erValidInterrupt;
    }

}
```

F. NAVIGATOR OF THE WATCH

```
package Thesis;

import simkit.*;
import simkit.random.*;
import java.util.*;

/**
 * The NavigatorOfTheWatch simulates a watch aboard the HSV-2. The NOOW listens
 * to the OfficerOfTheDeck for course and speed changes and the ContactGenerator
 * for new surface (SU) contacs.  NOWW also listens to the Interruption
 * Generator to implement MITIS.
 */
public class NavigatorOfTheWatch extends SimEntityBase {
    //parameters
    /**
     * random generator for times to update the deck log
     */
    private simkit.random.RandomVariate timeToUpdateDeckLogGen;
    /**
     * random generator for times to designate contacts on the ARPA
     */
    private simkit.random.RandomVariate timeToDesignateContactGen;
    /**
     * random generator for times to update the position log
     */
    private simkit.random.RandomVariate timeToUpdatePositionLogGen;
    /**
     * random generator for times to manage ECTDS
     */
    private simkit.random.RandomVariate timeToManageECTIDSGen;
    /**
     * double value for the current interval of time between fixes
     */
    private double fixInterval;

    //state variables
    /**
     * Integer 1 if NOOW is busy, 0 otherwise
     */
    protected int noowBusy;
    /**
     * true if in the middle of an interruption, false otherwise
     */
    protected boolean noowValidInterrupt;
    /**
     * work completed on a task at the time of an
     * interruption
```

```
     */
    protected double workDone;
    /**
     * time current task was started
     */
    protected double taskStartTime;
    /**
     * time left to complete a task
     */
    protected double taskLength;
    /**
     * time current interruption started
     */
    protected double interruptStartTime;
    /**
     * total time interrupted
     */
    protected double timeInterrupted;
    /**
     * queue of tasks waiting to be completed
     */
    protected      java.util.LinkedList      queue      =      new
java.util.LinkedList();

    //Event graph

    /**
     *Zero parameter constructor
     */
    public NavigatorOfTheWatch(){
    }

    /**
     * Creates a new instance of NavigatorOfTheWatch
     * @param ttudlg time to update deck log
     * @param ttdcg time to designate contact
     * @param ttuplg time to update position log
     * @param ttmectidsg time to manage ECTIDS
     * @param fi fix interval
     */
    public         NavigatorOfTheWatch(simkit.random.RandomVariate
ttudlg,
            simkit.random.RandomVariate ttdcg,
            simkit.random.RandomVariate ttuplg,
            simkit.random.RandomVariate ttmectidsg,
            double fi) {
        setTimeToUpdateDeckLogGen(ttudlg);
        setTimeToDesignateContactGen(ttdcg);
        setTimeToUpdatePositionLogGen(ttuplg);
        setTimeToManageECTIDSGen(ttmectidsg);
        setFixInterval(fi);
    }

    /** Set initial values of all state variables */
    public void reset() {

        super.reset();
```

```
        /** StateTransitions for the Run Event */

        this.noowBusy = 0;
        this.workDone = 0.0;
        this.taskStartTime = 0.0;
        this.taskLength = 0.0;
        this.queue.clear();
        this.noowValidInterrupt=false;
    }

    /**
     * initiates this class in the simulation model
     */
    public void doRun() {
        firePropertyChange("noowBusy",noowBusy);
        firePropertyChange("workDone",workDone);
        firePropertyChange("taskStartTime",taskStartTime);
        firePropertyChange("taskLength",taskLength);
        firePropertyChange("queue",queue);

firePropertyChange("noowValidInterrupt",noowValidInterrupt);
        if (true) {

waitDelay("UpdatePositionLogRequest",fixInterval,0.0);
        }
        if (true) {
            waitDelay("ManageECTIDSRequest",fixInterval,0.0);
        }
    }

    /**
     * Adds a BeginUpdatePositionLog to the task queue, Schedules
it if
     * the NOOW is not busy or not currently interrupted.
     */
    public void doUpdatePositionLogRequest() {
        queue.add(new String("BeginUpdatePositionLog"));
        firePropertyChange("queue", queue);
        waitDelay("UpdatePositionLogRequest",fixInterval,0.0);
        if (noowBusy==0 && !noowValidInterrupt) {
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            waitDelay("BeginUpdatePositionLog",0.0,new
Object[]{},1.0);
        }
    }

    /**
     * Adds a BeginManageECTIDS to the task queue, Schedules it
if
     * the NOOW is not busy or not currently interrupted.
     */
    public void doManageECTIDSRequest() {
        queue.add(new String("BeginManageECTIDS"));
        firePropertyChange("queue", queue);
        waitDelay("ManageECTIDSRequest",fixInterval,0.0);
```

```
        if (noowBusy==0 && !noowValidInterrupt) {
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            waitDelay("BeginManageECTIDS",0.0,1.0);
        }
    }

    /**
     * Removes task from event queue and scedules an
     * NOOWEndTask after a randomly generated task length.
     */
    public void doBeginUpdatePositionLog() {
        noowBusy = 1;
        firePropertyChange("noowBusy", noowBusy);
        queue.removeFirst();
        firePropertyChange("queue", queue);
        taskLength = timeToUpdatePositionLogGen.generate();
        firePropertyChange("taskLength", taskLength);
        taskStartTime = Schedule.getSimTime();
        firePropertyChange("taskStartTime", taskStartTime);
        waitDelay("NOOWEndTask",taskLength,0.0);
    }

    /**
     * Removes task from event queue and scedules an
     * NOOWEndTask after a randomly generated task length.
     */
    public void doBeginManageECTIDS() {
        noowBusy = 1;
        firePropertyChange("noowBusy", noowBusy);
        queue.removeFirst();
        firePropertyChange("queue", queue);
        taskLength = timeToManageECTIDSGen.generate();
        firePropertyChange("taskLength", taskLength);
        taskStartTime = Schedule.getSimTime();
        firePropertyChange("taskStartTime", taskStartTime);
        waitDelay("NOOWEndTask",taskLength,0.0);
    }

    /**
     * Schedules the next task in the event queue if queue is
     * not empty.
     */
    public void doNOOWEndTask() {
        noowBusy = 0;
        firePropertyChange("noowBusy", noowBusy);
        if(queue.size()>0){
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            String nextEvent = ((String)queue.getFirst());
            waitDelay(nextEvent,0.0,1.0);
        }
    }

    /**
     *  Adds  a  BeginUpdateDeckLog  request  to  the  task  queue,
Schedules it if
```

```
     * the NOOW is not busy or not currently interrupted. Adds a
Begin
     * MAnageECTIDS to the task queue.
     */
    public void doOODCourseChange() {
        queue.add(new String("BeginUpdateDeckLog"));
        queue.add(new String("BeginManageECTIDS"));
        firePropertyChange("queue", queue);
        if (noowBusy==0 && !noowValidInterrupt) {
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            waitDelay("BeginUpdateDeckLog",0.0,1.0);
        }
    }

    /**
     *  Adds  a  BeginUpdateDeckLog  request  to  the  task  queue,
Schedules it if
     * the NOOW is not busy or not currently interrupted. Adds a
Begin
     * MAnageECTIDS to the task queue.
     */
    public void doOODSpeedChange() {
        queue.add(new String("BeginUpdateDeckLog"));
        queue.add(new String("BeginManageECTIDS"));
        firePropertyChange("queue", queue);
        if (noowBusy==0 && !noowValidInterrupt) {
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            waitDelay("BeginUpdateDeckLog",0.0,1.0);
        }
    }

    /**
     * Removes task from event queue and scedules an
     * NOOWEndTask after a randomly generated task length.
     */
    public void doBeginUpdateDeckLog() {
        noowBusy = 1;
        firePropertyChange("noowBusy", noowBusy);
        queue.removeFirst();
        firePropertyChange("queue", queue);
        taskStartTime = Schedule.getSimTime();
        firePropertyChange("taskStartTime", taskStartTime);
        taskLength = timeToUpdateDeckLogGen.generate();
        firePropertyChange("taskLength", taskLength);
        waitDelay("NOOWEndTask",taskLength,0.0);
    }

    /**
     * Adds a BeginDesignateNewContact request to the task queue,
Schedules it if
     * the NOOW is not busy or not currently interrupted.
     */
    public void doNewSUContact() {
        queue.add(new String("BeginDesignateNewContact"));
        firePropertyChange("queue", queue);
```

```
        if (noowBusy==0 && !noowValidInterrupt) {
            noowBusy=1;
            firePropertyChange("noowBusy",noowBusy);
            waitDelay("BeginDesignateNewContact",0.0,1.0);
        }
    }

    /**
     * Removes task from event queue and scedules an
     * NOOWEndTask after a randomly generated task length.
     */
    public void doBeginDesignateNewContact() {
        noowBusy = 1;
        firePropertyChange("noowBusy", noowBusy);
        queue.removeFirst();
        firePropertyChange("queue", queue);
        taskStartTime = Schedule.getSimTime();
        firePropertyChange("taskStartTime", taskStartTime);
        taskLength = timeToDesignateContactGen.generate();
        firePropertyChange("taskLength", taskLength);
        waitDelay("NOOWEndTask",taskLength,0.0);
    }

    /**
     * Cancels the current event and keeps track of how much
     * work was completed.
     */
    public void doStartInterrupt() {

        if(noowBusy==1 && !noowValidInterrupt){
            interruptStartTime = Schedule.getSimTime();
            firePropertyChange("interruptStartTime",
interruptStartTime);
            workDone = Schedule.getSimTime()-taskStartTime;
            firePropertyChange("workDone", workDone);
            this.noowValidInterrupt=true;

firePropertyChange("noowValidinterrupt",noowValidInterrupt);
            interrupt("NOOWEndTask");
        }
    }

    /**
     * Reschedules the current event taking into account
     * the work aleady completed on it.
     */
    public void doEndInterrupt() {
        if(noowValidInterrupt){
            taskStartTime=Schedule.getSimTime();
            firePropertyChange("taskStartTime",taskStartTime);
            timeInterrupted        =        Schedule.getSimTime()-
interruptStartTime;
            firePropertyChange("timeInterrupted",
timeInterrupted);
            taskLength = taskLength-workDone;
            firePropertyChange("taskLength", taskLength);
            this.noowValidInterrupt=false;
```

```
firePropertyChange("noowValidInterrupt",noowValidInterrupt);
            waitDelay("NOOWEndTask",taskLength,0.0);
        }
    }

    //setters and getters for private parameters
    public                                                    void
setTimeToUpdateDeckLogGen(simkit.random.RandomVariate ttudlg) {
        timeToUpdateDeckLogGen = ttudlg;
    }
    public                                 simkit.random.RandomVariate
getTimeToUpdateDeckLogGen() {
        return timeToUpdateDeckLogGen;
    }
    public                                                    void
setTimeToDesignateContactGen(simkit.random.RandomVariate ttdcg) {
        timeToDesignateContactGen = ttdcg;
    }
    public                                 simkit.random.RandomVariate
getTimeToDesignateContactGen() {
        return timeToDesignateContactGen;
    }
    public                                                    void
setTimeToUpdatePositionLogGen(simkit.random.RandomVariate  ttuplg)
{
        timeToUpdatePositionLogGen = ttuplg;
    }
    public                                 simkit.random.RandomVariate
getTimeToUpdatePositionLogGen() {
        return timeToUpdatePositionLogGen;
    }
    public                                                    void
setTimeToManageECTIDSGen(simkit.random.RandomVariate   ttmectidsg)
{
        timeToManageECTIDSGen = ttmectidsg;
    }
    public  simkit.random.RandomVariate  getTimeToManageECTIDSGen()
{
        return timeToManageECTIDSGen;
    }
    public void setFixInterval(double fi) {
        fixInterval = fi;
    }
    public double getFixInterval() {
        return fixInterval;
    }

    //getters for protected state variables
    public int getNoowBusy() {
        return  noowBusy;
    }
    public double getWorkDone() {
        return  workDone;
    }
    public double getTaskStartTime() {
        return  taskStartTime;
```

```
    }
    public double getTaskLength() {
        return  taskLength;
    }
    public double getInterruptStartTime() {
        return  interruptStartTime;
    }
    public double getTimeInterrupted() {
        return  timeInterrupted;
    }
    public java.util.LinkedList getQueue() {
        return (java.util.LinkedList) queue.clone();
    }
    public boolean getNoowValidInterrupt(){
        return this.noowValidInterrupt;
    }
}
```

G. OFFICER OF THE DECK

```
package Thesis;

import simkit.*;
import simkit.random.*;
import java.util.*;

/**
 * The OfficerOfTheDeck class models the same watch in the HSV-2. The class
 * listens to the Command class as well as the TAO class for course and
 * speed changes.  It also listens to the Contact class for new SU contacts
 * as well as the InterruptionGenerator class to implement MITIs.
 */
public class OfficerOfTheDeck extends SimEntityBase {
    //Parameters
    /**
     * Random generator for times to desigante contacts on the ARPA.
     */
    private simkit.random.RandomVariate timeToDesignateContactGen;
    /**
     * Random generator for the times it takes for the OOD to manually change
     * the course of the ship.
     */
    private simkit.random.RandomVariate timeToChangeCourseGen;
    /**
     * Random generator for the time it takes to talk on another ship on the VHF
     * bridge to bridge radio.
     */
    private simkit.random.RandomVariate timeToCallShipGen;
    /**
     * Double probability that a course change is necessary for contact avoidance
     * when another ship is encountered.
     */
    private double probabilityCourseChangeNecessary;
    /**
     * Double probability that a speed change is necessary for contact avoidance
     * when another ship is encountered.
     */
    private double probabilitySpeedChangeNecessary;
    /**
     * Random generator for the times it takes the OOD to change the speed of the ship.
     */
    private simkit.random.RandomVariate timeToChangeSpeedGen;
    /**
```

```
           * Random variate used to draw a random number to determine
wether or not a course
           * and speed change is required when contacting another ship.
           */
          private Random rn;
          /**
           * Random generator for the time that it takes the OOD to
monitor the ship's
           * position at every fix.
           */
          private simkit.random.RandomVariate timeToMonitorPositionGen;
          /**
           * double interval between navigational fixes.
           */
          private double fixInterval;

          //State variables
          /**
           * Integer 1 id OOD is busy, 0 otherwise.
           */
          protected int oodBusy;
          /**
           * Boolean true if an MITI is in progress, false otherwise.
           */
          protected boolean validInterrupt;
          /**
           * Double amount of time that is completed in the current
task.
           */
          protected double workDone;
          /**
           * Double start time of the current task.
           */
          protected double taskStartTime;
          /**
           * Duoble time that it will take to complete the current
task.  Generated from
           * random variates.
           */
          protected double taskLength;
          /**
           * Double start time of the current MITI.
           */
          protected double interruptStartTime;
          /**
           *  Double value of the length of the most recent
interruption.
           */
          protected double timeInterrupted;
          /**
           * Queue of tasks.
           */
          protected     java.util.LinkedList     queue     =     new
java.util.LinkedList();

          // Event Graph
```

```
        /**
         *zero parameter constructor
         */
        public OfficerOfTheDeck(){
        }

        /**
         * Creates a new instance of OfficerOfTheDeck
         * @param ttEngageAutog time to engage autopilot
         * @param ttDesigContactg time to designate a contact
         * @param ttDisAutog time to disengage the autopilot
         * @param ttChangeCourseg time to change course
         * @param ttCallShipg time to call a ship on bridge to bridge
         * @param ttMonPositGen time to monitor the ships position
         * @param pccn probability a course change is necessry
         * @param pscn probability a speed change is necessary
         * @param fixInterval fix interval
         * @param ttChangeSpeedg time to change speed
         */
        public            OfficerOfTheDeck(simkit.random.RandomVariate
ttDesigContactg,
                simkit.random.RandomVariate ttChangeCourseg,
                simkit.random.RandomVariate ttCallShipg,
                RandomVariate ttMonPositGen,
                double pccn,
                double pscn,
                double fixInterval,
                simkit.random.RandomVariate ttChangeSpeedg) {

            setTimeToDesignateContactGen(ttDesigContactg);
            setTimeToChangeCourseGen(ttChangeCourseg);
            setTimeToCallShipGen(ttCallShipg);
            setTimeToMonitorPositionGen(ttMonPositGen);
            setProbabilityCourseChangeNecessary(pccn);
            setProbabilitySpeedChangeNecessary(pscn);
            setFixInterval(fixInterval);
            setTimeToChangeSpeedGen(ttChangeSpeedg);
        }

        /** Set initial values of all state variables */
        public void reset() {

            super.reset();

            /** StateTransitions for the Run Event */

            this.oodBusy = 0;
            this.workDone = 0.0;
            this.taskStartTime = 0.0;
            this.taskLength = 0.0;
            this.queue.clear();
            this.rn = new Random();
            this.validInterrupt=false;
        }

        /**
         * Initiates this class in the HSV2 model.
```

```
         */
        public void doRun() {
            firePropertyChange("oodBusy",oodBusy);
            firePropertyChange("workDone",workDone);
            firePropertyChange("taskStartTime",taskStartTime);
            firePropertyChange("taskLength",taskLength);
            firePropertyChange("queue",queue);
            firePropertyChange("validInterrupt",validInterrupt);
            waitDelay("MonitorPosition",fixInterval);

        }

        /**
         * Adds a BeginMonitorPosition to the task queue, Schedules
it if
         * the OOD is not busy or not currently interrupted.
         */
        public void doMonitorPosition() {
            queue.add(new String("BeginMonitorPosition"));
            firePropertyChange("queue", queue);
            waitDelay("MonitorPosition",fixInterval,0.0);
            if (this.oodBusy==0 && !validInterrupt) {
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
                waitDelay("BeginMonitorPosition",0.0,1.0);
            }
        }

        /**
         * Removes task from event queue and scedules an OODEndTask
after a randomly
         * generated task length.
         */
        public void doBeginMonitorPosition() {
            this.oodBusy = 1;
            firePropertyChange("oodBusy", oodBusy);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            this.taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            this.taskLength = timeToMonitorPositionGen.generate();;
            firePropertyChange("taskLength", taskLength);
            waitDelay("OODEndTask",taskLength,0.0);
        }


        /**
         * Adds a BeginDesignateContact to the task queue, Schedules
it if
         * the OOD is not busy or not currently interrupted.
         */
        public void doNewSUContact() {
            queue.add(new String("BeginDesignateContact"));
            firePropertyChange("queue", queue);
            if (this.oodBusy==0 && !validInterrupt) {
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
```

```
                waitDelay("BeginDesignateContact",0.0,1.0);
            }
        }

        /**
         * Schedules an OOD course change from the Command or TAO
class.
         */
        public void doCommandCourseChange() {
            waitDelay("OODCourseChange",0.0,0.0);
        }

        /**
         * Schedules an OOD speed change from the Command or TAO
class.
         */
        public void doCommandSpeedChange() {
            waitDelay("OODSpeedChange",0.0,0.0);
        }

        /**
         * Removes task from event queue and scedules an OODEndTask
after a randomly
         * generated task length.
         */
        public void doBeginDesignateContact() {
            double rnCourse = rn.nextDouble();
            double rnSpeed = rn.nextDouble();
            oodBusy = 1;
            firePropertyChange("oodBusy", oodBusy);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToDesignateContactGen.generate();
            firePropertyChange("taskLength", taskLength);
            waitDelay("OODEndTask",taskLength,0.0);
            if (rnCourse<probabilityCourseChangeNecessary) {
                waitDelay("OODCourseChange",taskLength,0.0);
            }
            if (rnSpeed<probabilitySpeedChangeNecessary) {
                waitDelay("OODSpeedChange",taskLength,0.0);
            }
            if      (rnCourse<probabilityCourseChangeNecessary      ||
rnSpeed<probabilitySpeedChangeNecessary) {
                waitDelay("MonitorBridgeToBridge",taskLength,0.0);
            }
        }

        /**
         * Schedules the next task in the event queue if queue is not
empty.
         */
        public void doOODEndTask() {
            oodBusy = 0;
            firePropertyChange("oodBusy", oodBusy);
            if (queue.size()>0){
```

```
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
                String nextTask = ((String)queue.getFirst());
                waitDelay(nextTask,0.0,1.0);
            }
        }

        /**
         * Adds a BeginCourseChange to the task queue, Schedules it
if
         * the OOD is not busy or not currently interrupted.
         */
        public void doOODCourseChange() {
            queue.add(new String("BeginCourseChange"));
            firePropertyChange("queue", queue);
            if (this.oodBusy==0 && !validInterrupt) {
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
                waitDelay("BeginCourseChange",0.0,1.0);
            }
        }

        /**
         * Adds a BeginSpeedChange to the task queue, Schedules it if
         * the OOD is not busy or not currently interrupted.
         */
        public void doOODSpeedChange() {
            queue.add(new String("BeginSpeedChange"));
            firePropertyChange("queue", queue);
            if (this.oodBusy==0 && !validInterrupt) {
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
                waitDelay("BeginSpeedChange",0.0,1.0);
            }
        }

        /**
         * Adds a BeginRadioShip to the task queue, Schedules it if
         * the OOD is not busy or not currently interrupted.
         */
        public void doMonitorBridgeToBridge() {
            queue.add(new String("BeginRadioShip"));
            firePropertyChange("queue", queue);
            if (this.oodBusy==0 && !validInterrupt) {
                oodBusy=1;
                firePropertyChange("oodBusy",oodBusy);
                waitDelay("BeginRadioShip",0.0,1.0);
            }
        }

        /**
         * Removes task from event queue and scedules an OODEndTask
after a randomly
         * generated task length.
         */
        public void doBeginCourseChange() {
            this.oodBusy = 1;
```

```
            firePropertyChange("oodBusy", oodBusy);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToChangeCourseGen.generate();
            firePropertyChange("taskLength", taskLength);
            waitDelay("OODEndTask",taskLength,0.0);
        }

        /**
         * Removes task from event queue and scedules an OODEndTask
after a randomly
         * generated task length.
         */
        public void doBeginSpeedChange() {
            this.oodBusy = 1;
            firePropertyChange("oodBusy", oodBusy);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToChangeSpeedGen.generate();
            firePropertyChange("taskLength", taskLength);
            waitDelay("OODEndTask",taskLength,0.0);
        }

        /**
         * Removes task from event queue and scedules an OODEndTask
after a randomly
         * generated task length.
         */
        public void doBeginRadioShip() {
            this.oodBusy = 1;
            firePropertyChange("oodBusy", oodBusy);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToCallShipGen.generate();
            firePropertyChange("taskLength", taskLength);
            waitDelay("OODEndTask",taskLength,0.0);
        }

        /**
         * Cancels the current event and keeps track of how much
         * work was completed.
         */
        public void doStartInterrupt() {
            if(oodBusy==1 && !validInterrupt){
                interruptStartTime = Schedule.getSimTime();
                firePropertyChange("interruptStartTime",
interruptStartTime);
                workDone = Schedule.getSimTime()-taskStartTime;
                firePropertyChange("workDone", workDone);
                this.validInterrupt=true;
                firePropertyChange("validInterrupt",validInterrupt);
```

```
                interrupt("OODEndTask");
            }
        }

        /**
         * Reschedules the current event taking into account
         * the work aleady completed on it.
         */
        public void doEndInterrupt() {
            if(validInterrupt){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeInterrupted        =        Schedule.getSimTime()-
interruptStartTime;
                firePropertyChange("timeInterrupted",
timeInterrupted);
                taskLength = taskLength-workDone;
                firePropertyChange("taskLength", taskLength);
                this.validInterrupt=false;
                firePropertyChange("validInterrupt",validInterrupt);
                waitDelay("OODEndTask",taskLength,0.0);
            }
        }

        //Setters and Getters for private Parameters
        public                                                    void
setTimeToDesignateContactGen(simkit.random.RandomVariate ttdcg) {
            timeToDesignateContactGen = ttdcg;
        }
        public                                simkit.random.RandomVariate
getTimeToDesignateContactGen() {
            return timeToDesignateContactGen;
        }
        public                                                    void
setTimeToChangeCourseGen(simkit.random.RandomVariate ttccg) {
            timeToChangeCourseGen = ttccg;
        }
        public  simkit.random.RandomVariate  getTimeToChangeCourseGen()
{
            return timeToChangeCourseGen;
        }
        public  void  setTimeToCallShipGen(simkit.random.RandomVariate
ttcsg) {
            timeToCallShipGen = ttcsg;
        }
        public simkit.random.RandomVariate getTimeToCallShipGen() {
            return timeToCallShipGen;
        }
        public  void  setProbabilityCourseChangeNecessary(double  pccn)
{
            probabilityCourseChangeNecessary = pccn;
        }
        public double getProbabilityCourseChangeNecessary() {
            return probabilityCourseChangeNecessary;
        }
        public void setProbabilitySpeedChangeNecessary(double pscn) {
            probabilitySpeedChangeNecessary = pscn;
```

```
        }
        public double getProbabilitySpeedChangeNecessary() {
            return probabilitySpeedChangeNecessary;
        }
        public                                                    void
setTimeToChangeSpeedGen(simkit.random.RandomVariate ttcsg) {
            timeToChangeSpeedGen = ttcsg;
        }
        public  simkit.random.RandomVariate  getTimeToChangeSpeedGen()
{
            return timeToChangeSpeedGen;
        }
        public                                                    void
setTimeToMonitorPositionGen(simkit.random.RandomVariate ttmpg) {
            timeToMonitorPositionGen = ttmpg;
        }
        public                                simkit.random.RandomVariate
getTimeToMonitorPositionGen() {
            return timeToMonitorPositionGen;
        }
        public void setFixInterval(double fi) {
            fixInterval = fi;
        }
        public double getFixInterval() {
            return fixInterval;
        }

        //Getters for protected State Variables
        public int getOodBusy() {
            return  oodBusy;
        }
        public double getWorkDone() {
            return  workDone;
        }
        public double getTaskStartTime() {
            return  taskStartTime;
        }
        public double getTaskLength() {
            return  taskLength;
        }
        public double getInterruptStartTime() {
            return  interruptStartTime;
        }
        public double getTimeInterrupted() {
            return  timeInterrupted;
        }
        public boolean getValidInterrupt(){
            return this.validInterrupt;
        }
        public java.util.LinkedList getQueue() {
            return (java.util.LinkedList) queue.clone();
        }
    }
```

H. TACTICAL ACTION OFFICER

```
package Thesis;

import simkit.*;
import simkit.random.*;
import java.util.*;

/**
 * Class to simulate the TacticalActionOfficer that may be a part
of the HSV-2
 * or LCS future watch structure.  Modeled after the TAO on a DDG
class. TAO
 *  listens  to  the  Command  class  for  strike  missions  and  the
Contact class for
 * "Other" contacts.
 */
public class TacticalActionOfficer extends SimEntityBase {
    //Parameters
    /**
     * Random generator for the time it takes to evalate a new
"other" contact
     */
    private simkit.random.RandomVariate timeToEvaluateContactGen;
    /**
     * Random generator for the time it takes to talk on a HF
circuit such as
     * SATHICOM or BGCMD.
     */
    private simkit.random.RandomVariate timeToTalkOnRadioGen;
    /**
     *  Random  generator  for  the  time  it  takes  to  review  an
engagement of a contact
     * of interest, will vary according to mission being modeled.
     */
    private                              simkit.random.RandomVariate
timeToReviewEngagementGen;
    /**
     *  Random  generator  for  the  time  it  takes  to  monitor  the
GCCSM system.
     */
    private simkit.random.RandomVariate timeToMonitorGCCSGen;
    /**
     *  Random  generator  for  the  time  it  takes  to  moniotor  the
ships position at each
     * fix interval.
     */
    private simkit.random.RandomVariate timeToMonitorPositionGen;
    /**
     *  Random  generator  for  the  time  it  takes  to  plan  a  strike
mission.
     */
    private                              simkit.random.RandomVariate
timeToPlanStrikeMissionGen;
    /**
     * Random generator for the time between radio calls.
```

```
     */
    private simkit.random.RandomVariate radioIntervalGen;
    /**
     * probability that a new "Other" contact will be engaged. How hostile is the
     * environment?
     */
    private double probabilityEngageContact;
    /**
     * Random variate to draw a random number that will determine engagability.
     */
    private java.util.Random rn;
    /**
     * Fix interval.
     */
    private double fixInterval;

    //State Variables
    /**
     * Integer 1 if TAO is busy, 0 otherwise.
     */
    protected int busy;
    /**
     * Boolean true if a MITI is in progress, false otherwise.
     */
    protected boolean taoValidInterrupt;
    /**
     * The amount of work that has been completed on the current task.
     */
    protected double workDone;
    /**
     * The time that the current task was started.
     */
    protected double taskStartTime;
    /**
     * Randomly generated amount of time that the current task will take.
     */
    protected double taskLength;
    /**
     * Time that the current interruption started.
     */
    protected double interruptStartTime;
    /**
     * Total time of the current MITI.
     */
    protected double timeInterrupted;
    /**
     * Queue of tasks that must be completed.
     */
    protected java.util.LinkedList queue = new java.util.LinkedList();
    /**
     * The next task in the event queue.
     */
```

```
        protected String nextTask;

        //Event Graph

        /**
         *zero parameter constructor
         */
        public TacticalActionOfficer(){
        }

        /**
         * Creates a new instance of TacticalActionOfficer
         * @param ttecg time to evaluate contact
         * @param tttorg time to talk on HF radio
         * @param ttreg time to review engagement
         * @param ttmgccsg time to monitor GCCSM
         * @param ttmpg time to monitor position
         * @param ttpsmg time to plan strike mission
         * @param rig radio interval generator
         * @param pec probability of engaging a contact
         * @param fi fix interval
         */
        public       TacticalActionOfficer(simkit.random.RandomVariate
ttecg,
                simkit.random.RandomVariate tttorg,
                simkit.random.RandomVariate ttreg,
                simkit.random.RandomVariate ttmgccsg,
                simkit.random.RandomVariate ttmpg,
                simkit.random.RandomVariate ttpsmg,
                RandomVariate rig,
                double pec,
                double fi) {

            setTimeToEvaluateContactGen(ttecg);
            setTimeToTalkOnRadioGen(tttorg);
            setTimeToReviewEngagementGen(ttreg);
            setTimeToMonitorGCCSGen(ttmgccsg);
            setTimeToMonitorPositionGen(ttmpg);
            setProbabilityEngageContact(pec);
            setFixInterval(fi);
            setRadioIntervalGen(rig);
            setTimeToPlanStrikeMissionGen(ttpsmg);
        }

        /** Set initial values of all state variables */
        public void reset() {

            super.reset();

            /** StateTransitions for the Run Event */

            this.busy = 0;
            this.workDone = 0.0;
            this.taskStartTime = 0.0;
            this.taskLength = 0.0;
            this.queue.clear();
            this.rn = new Random();
```

```
            this.taoValidInterrupt=false;
        }

        /**
         * Initializes this class in the HSV2 model
         */
        public void doRun() {
            firePropertyChange("busy",busy);
            firePropertyChange("workDone",workDone);
            firePropertyChange("taskStartTime",taskStartTime);
            firePropertyChange("taskLength",taskLength);
            firePropertyChange("queue",queue);

firePropertyChange("taoValidInterrupt",taoValidInterrupt);
            waitDelay("TalkOnRadio",radioIntervalGen.generate(),0.0);
            waitDelay("MonitorGCCSM",30.0,0.0);
            waitDelay("MonitorShipsPosition",fixInterval,0.0);
        }

        /**
         * Schedules the next task in the event queue if queue is not
empty.
         */
        public void doTAOEndTask() {
            busy = 0;
            firePropertyChange("busy", busy);
            if(queue.size()>0){
                busy=1;
                firePropertyChange("busy",busy);
                String nextEvent = ((String)queue.getFirst());
                waitDelay(nextEvent,0.0,1.0);
            }
        }

        /**
         * Adds a BeginTalkOnRadio to the task queue, Schedules it if
         *  the  TAO  is  not  busy  or  not  currently  interrupted.
Sxhedules another
         * TalkOnRadio after a randomly generated time.
         */
        public void doTalkOnRadio() {
            queue.add(new String("BeginTalkOnRadio"));
            firePropertyChange("queue", queue);
            waitDelay("TalkOnRadio",radioIntervalGen.generate(),0.0);
            if (busy==0 && !taoValidInterrupt) {
                busy=1;
                firePropertyChange("busy",busy);
                waitDelay("BeginTalkOnRadio",0.0,1.0);
            }
        }

        /**
         * Adds a BeginMonitorGCCSM to the task queue, Schedules it
if
         *  the  TAO  is  not  busy  or  not  currently  interrupted.
Schedules
         * another MonitorGCCSM after a randomly generated time.
```

```
         */
        public void doMonitorGCCSM() {
            queue.add(new String("BeginMonitorGCCSM"));
            firePropertyChange("queue", queue);
            waitDelay("MonitorGCCSM",30.00,0.0);
            if (busy==0 && !taoValidInterrupt) {
                busy=1;
                firePropertyChange("busy",busy);
                waitDelay("BeginMonitorGCCSM",0.0,1.0);
            }
        }

        /**
         *  Adds  a  BeginMonitorShipsPosition  to  the  task  queue,
Schedules it if
         *  the  TAO  is  not  busy  or  not  currently  interrupted.
Schedules
         *  another  MonitorShipsPosition  after  a  randomly  generated
time.
         */
        public void doMonitorShipsPosition() {
            queue.add(new String("BeginMonitorShipsPosition"));
            firePropertyChange("queue", queue);
            waitDelay("MonitorShipsPosition",fixInterval,new
Object[]{},0);
            if (busy==0 && !taoValidInterrupt) {
                busy=1;
                firePropertyChange("busy",busy);
                waitDelay("BeginMonitorShipsPosition",0.0,1.0);
            }
        }

        /**
         * Removes task from event queue and scedules an
         * TAOEndTask after a randomly generated task length.
         */
        public void doBeginMonitorGCCSM() {
            busy = 1;
            firePropertyChange("busy", busy);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToMonitorGCCSGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            nextTask = "TAOEndTask";
            firePropertyChange("nextTask", nextTask);
            waitDelay("TAOEndTask",taskLength,0.0);
        }

        /**
         * Removes task from event queue and scedules an
         * TAOEndTask after a randomly generated task length.
         */
        public void doBeginMonitorShipsPosition() {
            busy = 1;
            firePropertyChange("busy", busy);
```

```
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToMonitorPositionGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            nextTask = "TAOEndTask";
            firePropertyChange("nextTask", nextTask);
            waitDelay("TAOEndTask",taskLength,0.0);
        }

        /**
         * Removes task from event queue and scedules an
         * TAOEndTask after a randomly generated task length.
         */
        public void doBeginTalkOnRadio() {
            busy = 1;
            firePropertyChange("busy", busy);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToTalkOnRadioGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            nextTask = "TAOEndTask";
            firePropertyChange("nextTask", nextTask);
            waitDelay("TAOEndTask",taskLength,0.0);
        }

        /**
         * Removes task from event queue and scedules an
         * TAOEndTask after a randomly generated task length.
         */
        public void doBeginReviewEngagement() {
            busy = 1;
            firePropertyChange("busy", busy);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToReviewEngagementGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            nextTask = "TAOEndTask";
            firePropertyChange("nextTask", nextTask);
            waitDelay("TAOEndTask",taskLength,0.0);
        }

        /**
         * Adds a BeginEvaluateContact to the task queue, Schedules
it if
         * the TAO is not busy or not currently interrupted.
         */
        public void doNewContact() {
            queue.add(new String("BeginEvaluateContact"));
            firePropertyChange("queue", queue);
            if (busy==0 && !taoValidInterrupt) {
                busy=1;
```

```
                firePropertyChange("busy",busy);
                waitDelay("BeginEvaluateContact",0.0,1.0);
            }
        }

        /**
         * Adds a PlanNewStrikeMission to the task queue, Schedules
it if
         * the TAO is not busy or not currently interrupted.
         */
        public void doNewStrikeMission() {
            queue.add(new String("PlanNewStrikeMission"));
            firePropertyChange("queue", queue);
            if (busy==0 && !taoValidInterrupt) {
                busy=1;
                firePropertyChange("busy",busy);
                waitDelay("PlanNewStrikeMission",0.0,1.0);
            }
        }

        /**
         * Removes task from event queue and scedules an
         * TAOEndTask after a randomly generated task length.
         */
        public void doBeginEvaluateContact() {
            double engageable = rn.nextDouble();
            busy = 1;
            firePropertyChange("busy", busy);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToEvaluateContactGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            if (engageable<probabilityEngageContact) {
                nextTask = "EngageContact";
                firePropertyChange("nextTask", nextTask);
                waitDelay("EngageContact",taskLength,1.0);
            }
            else {
                nextTask = "TAOEndTask";
                firePropertyChange("nextTask",nextTask);
                waitDelay("TAOEndTask",taskLength,0.0);
            }
        }

        /**
         * Adds a BeginTalkOnRadio and a BeginReviewEngagement to the
front of the task
         *  queue.   Schedules an immediate CommandCourseChange and
CommandSpeedChange as
         *  well as an TAOEndTask to begin scheduling the most
recently added engagement
         * event.
         */
        public void doEngageContact() {
            waitDelay("CommandCourseChange",0.0,0.0);
```

```
            waitDelay("CommandSpeedChange",0.0,0.0);
            queue.addFirst(new String("BeginReviewEngagement"));
            firePropertyChange("queue", queue);
            queue.addFirst(new String("BeginTalkOnRadio"));
            firePropertyChange("queue", queue);
            waitDelay("TAOEndTask",0.0,1.0);
        }

        /**
         * Schedules an EngageContact after a randomly generated task
length.
         */
        public void doPlanNewStrikeMission() {
            busy = 1;
            firePropertyChange("busy", busy);
            taskStartTime = Schedule.getSimTime();
            firePropertyChange("taskStartTime", taskStartTime);
            taskLength = timeToPlanStrikeMissionGen.generate();
            firePropertyChange("taskLength", taskLength);
            queue.removeFirst();
            firePropertyChange("queue", queue);
            nextTask = "EngageContact";
            firePropertyChange("nextTask", nextTask);
            waitDelay("EngageContact",taskLength,1.0);
        }

        /**
         * Cancels the current event and keeps track of how much
         * work was completed.
         */
        public void doStartInterrupt() {
            if(busy==1 && !taoValidInterrupt){
                interruptStartTime = Schedule.getSimTime();
                firePropertyChange("interruptStartTime",
interruptStartTime);
                workDone = Schedule.getSimTime()-taskStartTime;
                firePropertyChange("workDone", workDone);
                this.taoValidInterrupt=true;

firePropertyChange("taoValidInterrupt",taoValidInterrupt);
                interrupt(nextTask);
            }
        }

        /**
         * Reschedules the current event taking into account
         * the work aleady completed on it.
         */
        public void doEndInterrupt() {
            if(taoValidInterrupt){
                taskStartTime=Schedule.getSimTime();
                firePropertyChange("taskStartTime",taskStartTime);
                timeInterrupted        =        Schedule.getSimTime()-
interruptStartTime;
                firePropertyChange("timeInterrupted",
timeInterrupted);
                taskLength = taskLength-workDone;
```

```
                firePropertyChange("taskLength", taskLength);
                this.taoValidInterrupt=false;

firePropertyChange("taoValidInterrupt",taoValidInterrupt);
                waitDelay(nextTask,taskLength,0.0);
            }
        }

        //Setters and Getters for private parameters
        public                                                  void
setTimeToEvaluateContactGen(simkit.random.RandomVariate ttecg) {
            timeToEvaluateContactGen = ttecg;
        }
        public                              simkit.random.RandomVariate
getTimeToEvaluateContactGen() {
            return timeToEvaluateContactGen;
        }
        public                                                  void
setTimeToTalkOnRadioGen(simkit.random.RandomVariate tttorg) {
            timeToTalkOnRadioGen = tttorg;
        }
        public  simkit.random.RandomVariate  getTimeToTalkOnRadioGen()
{
            return timeToTalkOnRadioGen;
        }
        public                                                  void
setTimeToReviewEngagementGen(simkit.random.RandomVariate ttreg) {
            timeToReviewEngagementGen = ttreg;
        }
        public                              simkit.random.RandomVariate
getTimeToReviewEngagementGen() {
            return timeToReviewEngagementGen;
        }
        public                                                  void
setTimeToMonitorGCCSGen(simkit.random.RandomVariate ttmgccsg) {
            timeToMonitorGCCSGen = ttmgccsg;
        }
        public  simkit.random.RandomVariate  getTimeToMonitorGCCSGen()
{
            return timeToMonitorGCCSGen;
        }
        public                                                  void
setTimeToMonitorPositionGen(simkit.random.RandomVariate ttmpg) {
            timeToMonitorPositionGen = ttmpg;
        }
        public                              simkit.random.RandomVariate
getTimeToMonitorPositionGen() {
            return timeToMonitorPositionGen;
        }
        public void setProbabilityEngageContact(double pec) {
            probabilityEngageContact = pec;
        }
        public double getProbabilityEngageContact() {
            return probabilityEngageContact;
        }
        public void setFixInterval(double fi) {
            fixInterval = fi;
```

```
        }
        public double getFixInterval() {
            return fixInterval;
        }
        public void setRadioIntervalGen(RandomVariate rig) {
            radioIntervalGen = rig;
        }
        public RandomVariate getRadioIntervalGen() {
            return radioIntervalGen;
        }
        public                                                    void
setTimeToPlanStrikeMissionGen(simkit.random.RandomVariate ttpsmg) {
            timeToPlanStrikeMissionGen = ttpsmg;
        }
        public                                simkit.random.RandomVariate
getTimeToPlanStrikeMissionGen() {
            return timeToPlanStrikeMissionGen;
        }

        //Getters for protected State variables
        public int getBusy() {
            return  busy;
        }
        public double getWorkDone() {
            return  workDone;
        }
        public double getTaskStartTime() {
            return  taskStartTime;
        }
        public double getTaskLength() {
            return  taskLength;
        }
        public double getInterruptStartTime() {
            return  interruptStartTime;
        }
        public double getTimeInterrupted() {
            return  timeInterrupted;
        }
        public java.util.LinkedList getQueue() {
            return (java.util.LinkedList) queue.clone();
        }
        public String getNextTask() {
            return  nextTask;
        }
        public boolean getTaoValidInterrupt(){
            return this.taoValidInterrupt;
        }

    }
```

APPENDIX F. SIMULATION INPUT DATA

A. NOLH DESIGN AND CONSTANT SIMULATION PARAMETERS

1. Engineering Rover Simulation Run

all values are in minutes

Simulation Entity	Parameter Name	Description	Constant Parameter Values	NOHL Design Values	
				Min	Max
Assembly:	Sim.Run.Time	Double	**40000.00**		
	Nbr.Repititions	Integer	**10**		
	Fix.Interval	Double		**15.00**	**60.00**
ContactGenerator:	Other.IA.Time	Exp(μ)	**INF**		
	SU.IA.Time	Exp(μ)		**15.00**	**120.00**
InterruptionGenerator:	Time.btw.Interrupt	Exp(μ)		**0.88**	**60.00**
	Interrupt.Length	Gamma(α,β); μ		**0.04**	**0.41**
		σ		**0.01**	**0.17**
EngineeringRover:	Time.btwn.Rnds	Exp(μ)	**45.00**		
	Short.Xsit.Time	Gamma(α,β); μ	**1.18**		
		σ	**0.14**		
	Long.Xsit.Time	Gamma(α,β); μ	**3.10**		
		σ	**0.25**		
Command:	Time.btwn.Crse.Change	Exp(μ)		**60.00**	**240.00**
	Time.btwn.Spd.Change	Exp(μ)		**60.00**	**240.00**
	Time.btwn.Strike.Missn	Exp(μ)	**INF**		
NavigatorOfTheWatch:	Time.Updte.Deck.Log	Gamma(α,β); μ	**0.24**		
		σ	**0.06**		
	Time.Dsg.Contact	Gamma(α,β); μ	**0.33**		
		σ	**0.05**		
	Time.Updte.Posit.Log	Gamma(α,β); μ	**0.53**		
		σ	**0.04**		
	Time.Mnge.ECTDS	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
OfficerOfTheDeck	Prob.Crse.Change	Double		**0.05**	**0.50**
	Prob.Spd.Change	Double		**0.00**	**0.38**
	Time.Change.Crse	Gamma(α,β); μ	**0.42**		
		σ	**0.22**		
	Time.Call.Ship	Gamma(α,β); μ	**0.91**		
		σ	**0.25**		
	Time.Change.Spd	Gamma(α,β); μ	**0.20**		
		σ	**0.04**		
	Time.Mon.Posit	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
TacticalActionOfficer	Time.Eval.Contact	Gamma(α,β); μ	**0.57**		
		σ	**0.50**		
	Time.Talk.Radio	Gamma(α,β); μ	**0.63**		
		σ	**1.16**		
	Time.Review.Engage	Gamma(α,β); μ	**1.36**		
		σ	**1.27**		
	Time.Mon.GCCS	Gamma(α,β); μ	**0.10**		
		σ	**0.07**		
	Time.Plan.Strike	Gamma(α,β); μ	**0.77**		
		σ	**0.28**		
	Radio.Interval	Exp(μ)		**30.00**	**60.00**
	Prob.Engagage.Contact	Double	**0.00**		

2. Open Ocean Transit Simulation Run

all values are in minutes

Simulation Entity	Parameter Name	Description	Constant Parameter Values	NOHL Design Values Min	NOHL Design Values Max
Assembly:	Sim.Run.Time	Double	**40000.00**		
	Nbr.Repititions	Integer	**10**		
	Fix.Interval	Double		**15.00**	**60.00**
ContactGenerator:	Other.IA.Time	Exp(μ)	**INF**		
	SU.IA.Time	Exp(μ)		**15.00**	**120.00**
InterruptionGenerator:	Time.btw.Interrupt	Exp(μ)		**0.88**	**60.00**
	Interrupt.Length	Gamma(α,β); μ		**0.10**	**1.10**
		σ		**0.04**	**0.44**
EngineeringRover:	Time.btwn.Rnds	Exp(μ)	**45.00**		
	Short.Xsit.Time	Gamma(α,β); μ	**1.18**		
		σ	**0.14**		
	Long.Xsit.Time	Gamma(α,β); μ	**3.10**		
		σ	**0.25**		
Command:	Time.btwn.Crse.Change	Exp(μ)		**60.00**	**240.00**
	Time.btwn.Spd.Change	Exp(μ)		**60.00**	**240.00**
	Time.btwn.Strike.Missn	Exp(μ)	**INF**		
NavigatorOfTheWatch:	Time.Updte.Deck.Log	Gamma(α,β); μ	**0.24**		
		σ	**0.06**		
	Time.Dsg.Contact	Gamma(α,β); μ	**0.33**		
		σ	**0.05**		
	Time.Updte.Posit.Log	Gamma(α,β); μ	**0.53**		
		σ	**0.04**		
	Time.Mnge.ECTDS	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
OfficerOfTheDeck	Prob.Crse.Change	Double		**0.05**	**0.50**
	Prob.Spd.Change	Double		**0.00**	**0.38**
	Time.Change.Crse	Gamma(α,β); μ	**0.42**		
		σ	**0.22**		
	Time.Call.Ship	Gamma(α,β); μ	**0.91**		
		σ	**0.25**		
	Time.Change.Spd	Gamma(α,β); μ	**0.20**		
		σ	**0.04**		
	Time.Mon.Posit	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
TacticalActionOfficer	Time.Eval.Contact	Gamma(α,β); μ	**0.57**		
		σ	**0.50**		
	Time.Talk.Radio	Gamma(α,β); μ	**0.63**		
		σ	**1.16**		
	Time.Review.Engage	Gamma(α,β); μ	**1.36**		
		σ	**1.27**		
	Time.Mon.GCCS	Gamma(α,β); μ	**0.10**		
		σ	**0.07**		
	Time.Plan.Strike	Gamma(α,β); μ	**0.77**		
		σ	**0.28**		
	Radio.Interval	Exp(μ)		**30.00**	**60.00**
	Prob.Engagage.Contact	Double	**0.00**		

3. Littoral Transit Simulation Run

all values are in minutes

Simulation Entity	Parameter Name	Description	Constant Parameter Values	NOHL Design Values Min	NOHL Design Values Max
Assembly:	Sim.Run.Time	Double	**40000.00**		
	Nbr.Repititions	Integer	**10**		
	Fix.Interval	Double		**3.00**	**30.00**
ContactGenerator:	Other.IA.Time	Exp(μ)	**INF**		
	SU.IA.Time	Exp(μ)		**2.00**	**15.00**
InterruptionGenerator:	Time.btw.Interrupt	Exp(μ)		**0.88**	**60.00**
	Interrupt.Length	Gamma(α,β); μ		**0.10**	**1.10**
		σ		**0.04**	**0.44**
EngineeringRover:	Time.btwn.Rnds	Exp(μ)	**45.00**		
	Short.Xsit.Time	Gamma(α,β); μ	**1.18**		
		σ	**0.14**		
	Long.Xsit.Time	Gamma(α,β); μ	**3.10**		
		σ	**0.25**		
Command:	Time.btwn.Crse.Change	Exp(μ)		**5.00**	**40.00**
	Time.btwn.Spd.Change	Exp(μ)		**5.00**	**60.00**
	Time.btwn.Strike.Missn	Exp(μ)	**INF**		
NavigatorOfTheWatch:	Time.Updte.Deck.Log	Gamma(α,β); μ	**0.24**		
		σ	**0.06**		
	Time.Dsg.Contact	Gamma(α,β); μ	**0.33**		
		σ	**0.05**		
	Time.Updte.Posit.Log	Gamma(α,β); μ	**0.53**		
		σ	**0.04**		
	Time.Mnge.ECTDS	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
OfficerOfTheDeck	Prob.Crse.Change	Double		**0.00**	**0.95**
	Prob.Spd.Change	Double		**0.05**	**0.90**
	Time.Change.Crse	Gamma(α,β); μ	**0.42**		
		σ	**0.22**		
	Time.Call.Ship	Gamma(α,β); μ	**0.91**		
		σ	**0.25**		
	Time.Change.Spd	Gamma(α,β); μ	**0.20**		
		σ	**0.04**		
	Time.Mon.Posit	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
TacticalActionOfficer	Time.Eval.Contact	Gamma(α,β); μ	**0.57**		
		σ	**0.50**		
	Time.Talk.Radio	Gamma(α,β); μ	**0.63**		
		σ	**1.16**		
	Time.Review.Engage	Gamma(α,β); μ	**1.36**		
		σ	**1.27**		
	Time.Mon.GCCS	Gamma(α,β); μ	**0.10**		
		σ	**0.07**		
	Time.Plan.Strike	Gamma(α,β); μ	**0.77**		
		σ	**0.28**		
	Radio.Interval	Exp(μ)		**5.00**	**30.00**
	Prob.Engagage.Contact	Double	**0.00**		

4. Anti Surface Warfare Simulation Run

all values are in minutes

Simulation Entity	Parameter Name	Description	Constant Parameter Values	NOHL Design Values Min	NOHL Design Values Max
Assembly:	Sim.Run.Time	Double	**40000.00**		
	Nbr.Repititions	Integer	**10**		
	Fix.Interval	Double		**3.00**	**30.00**
ContactGenerator:	Other.IA.Time	Exp(μ)	**INF**		
	SU.IA.Time	Exp(μ)		**2.00**	**15.00**
InterruptionGenerator:	Time.btw.Interrupt	Exp(μ)		**0.88**	**60.00**
	Interrupt.Length	Gamma(α,β); μ		**0.10**	**1.10**
		σ		**0.04**	**0.44**
EngineeringRover:	Time.btwn.Rnds	Exp(μ)	**45.00**		
	Short.Xsit.Time	Gamma(α,β); μ	**1.18**		
		σ	**0.14**		
	Long.Xsit.Time	Gamma(α,β); μ	**3.10**		
		σ	**0.25**		
Command:	Time.btwn.Crse.Change	Exp(μ)		**5.00**	**15.00**
	Time.btwn.Spd.Change	Exp(μ)		**5.00**	**15.00**
	Time.btwn.Strike.Missn	Exp(μ)	**INF**		
NavigatorOfTheWatch:	Time.Updte.Deck.Log	Gamma(α,β); μ	**0.24**		
		σ	**0.06**		
	Time.Dsg.Contact	Gamma(α,β); μ	**0.33**		
		σ	**0.05**		
	Time.Updte.Posit.Log	Gamma(α,β); μ	**0.53**		
		σ	**0.04**		
	Time.Mnge.ECTDS	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
OfficerOfTheDeck	Prob.Crse.Change	Double		**0.05**	**0.50**
	Prob.Spd.Change	Double		**0.00**	**0.38**
	Time.Change.Crse	Gamma(α,β); μ	**0.42**		
		σ	**0.22**		
	Time.Call.Ship	Gamma(α,β); μ	**0.91**		
		σ	**0.25**		
	Time.Change.Spd	Gamma(α,β); μ	**0.20**		
		σ	**0.04**		
	Time.Mon.Posit	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
TacticalActionOfficer	Time.Eval.Contact	Gamma(α,β); μ	**0.57**		
		σ	**0.50**		
	Time.Talk.Radio	Gamma(α,β); μ	**0.63**		
		σ	**1.16**		
	Time.Review.Engage	Gamma(α,β); μ	**1.36**		
		σ	**1.27**		
	Time.Mon.GCCS	Gamma(α,β); μ	**0.10**		
		σ	**0.07**		
	Time.Plan.Strike	Gamma(α,β); μ	**0.77**		
		σ	**0.28**		
	Radio.Interval	Exp(μ)		**5.00**	**30.00**
	Prob.Engagage.Contact	Double		**0.10**	**1.00**

5. Naval Surface Fire Support Simulation Run

all values are in minutes

Simulation Entity	Parameter Name	Description	Constant Parameter Values	NOHL Design Values Min	NOHL Design Values Max
Assembly:	Sim.Run.Time	Double	**40000.00**		
	Nbr.Repititions	Integer	**10**		
	Fix.Interval	Double		**3.00**	**15.00**
ContactGenerator:	Other.IA.Time	Exp(μ)	**INF**		
	SU.IA.Time	Exp(μ)		**5.00**	**45.00**
InterruptionGenerator:	Time.btw.Interrupt	Exp(μ)		**0.88**	**60.00**
	Interrupt.Length	Gamma(α,β); μ		**0.10**	**1.10**
		σ		**0.04**	**0.44**
EngineeringRover:	Time.btwn.Rnds	Exp(μ)	**45.00**		
	Short.Xsit.Time	Gamma(α,β); μ	**1.18**		
		σ	**0.14**		
	Long.Xsit.Time	Gamma(α,β); μ	**3.10**		
		σ	**0.25**		
Command:	Time.btwn.Crse.Change	Exp(μ)		**5.00**	**40.00**
	Time.btwn.Spd.Change	Exp(μ)		**5.00**	**60.00**
	Time.btwn.Strike.Missn	Exp(μ)		**2.00**	**30.00**
NavigatorOfTheWatch:	Time.Updte.Deck.Log	Gamma(α,β); μ	**0.24**		
		σ	**0.06**		
	Time.Dsg.Contact	Gamma(α,β); μ	**0.33**		
		σ	**0.05**		
	Time.Updte.Posit.Log	Gamma(α,β); μ	**0.53**		
		σ	**0.04**		
	Time.Mnge.ECTDS	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
OfficerOfTheDeck	Prob.Crse.Change	Double		**0.05**	**0.50**
	Prob.Spd.Change	Double		**0.00**	**0.38**
	Time.Change.Crse	Gamma(α,β); μ	**0.42**		
		σ	**0.22**		
	Time.Call.Ship	Gamma(α,β); μ	**0.91**		
		σ	**0.25**		
	Time.Change.Spd	Gamma(α,β); μ	**0.20**		
		σ	**0.04**		
	Time.Mon.Posit	Gamma(α,β); μ	**0.10**		
		σ	**0.03**		
TacticalActionOfficer	Time.Eval.Contact	Gamma(α,β); μ	**0.57**		
		σ	**0.50**		
	Time.Talk.Radio	Gamma(α,β); μ	**0.63**		
		σ	**1.16**		
	Time.Review.Engage	Gamma(α,β); μ	**1.36**		
		σ	**1.27**		
	Time.Mon.GCCS	Gamma(α,β); μ	**0.10**		
		σ	**0.07**		
	Time.Plan.Strike	Gamma(α,β); μ	**0.77**		
		σ	**0.28**		
	Radio.Interval	Exp(μ)			
	Prob.Engagage.Contact	Double	**0.00**		

B. SIMULATION INPUTS FROM NOLH DESIGN VARIABLES

These are the NOLH generated parameter values between the minimum and maximum values for each parameter. All of the other parameters remain constant for each run and are as listed in part 1 above.

1. Engineering Rover Simulation Run

Param. Grp.	TimeBtwnInterrupt	IntLengthMean	InterLengthSigma
1.00	22.13	0.16	0.03
2.00	7.35	0.20	0.07
3.00	57.23	0.12	0.06
4.00	43.37	0.21	0.02
5.00	11.97	0.05	0.03
6.00	14.74	0.22	0.04
7.00	32.29	0.05	0.05
8.00	55.38	0.16	0.07
9.00	1.80	0.34	0.08
10.00	29.52	0.32	0.01
11.00	58.15	0.24	0.04
12.00	41.53	0.38	0.04
13.00	20.28	0.28	0.07
14.00	25.82	0.37	0.05
15.00	47.07	0.39	0.08
16.00	33.21	0.31	0.03
17.00	23.97	0.09	0.12
18.00	16.58	0.17	0.10
19.00	34.14	0.15	0.14
20.00	51.69	0.06	0.10
21.00	24.90	0.07	0.17
22.00	0.88	0.10	0.09
23.00	47.99	0.13	0.17
24.00	49.84	0.18	0.12
25.00	6.42	0.26	0.14
26.00	10.12	0.41	0.13
27.00	45.22	0.33	0.10
28.00	37.83	0.35	0.16
29.00	8.27	0.27	0.11
30.00	21.20	0.31	0.15
31.00	42.45	0.25	0.16
32.00	56.31	0.36	0.13
33.00	30.44	0.23	0.09
34.00	38.75	0.29	0.15
35.00	53.53	0.25	0.12
36.00	3.65	0.33	0.12
37.00	17.51	0.24	0.16
38.00	48.92	0.40	0.16
39.00	46.14	0.23	0.15
40.00	28.59	0.40	0.13
41.00	5.50	0.29	0.11
42.00	59.08	0.11	0.11
43.00	31.36	0.13	0.17
44.00	2.73	0.21	0.14
45.00	19.36	0.07	0.14
46.00	40.60	0.17	0.11
47.00	35.06	0.08	0.13
48.00	13.81	0.06	0.10
49.00	27.67	0.14	0.15
50.00	36.91	0.36	0.06
51.00	44.30	0.28	0.08
52.00	26.75	0.30	0.05
53.00	9.19	0.39	0.08
54.00	35.98	0.38	0.01
55.00	60.00	0.35	0.09
56.00	12.89	0.32	0.02
57.00	11.04	0.27	0.06
58.00	54.46	0.19	0.04
59.00	50.76	0.04	0.05
60.00	15.66	0.12	0.09
61.00	23.05	0.10	0.02
62.00	52.61	0.18	0.07
63.00	39.68	0.14	0.03
64.00	18.43	0.20	0.02
65.00	4.58	0.09	0.06

2. Open Ocean Transit Simulation Run

Param. Grp.	fix interval	SU_IATime	TimeBtwnInterrupt	IntLengthMean	InterLengthSigma	TimeBtwnCrseChg	TimeBtwnSpdChg	PrCrseChg	PrSpdChg	RadioInt
1.00	47.00	20.00	22.13	0.43	0.09	198.00	203.00	0.27	0.37	52.00
2.00	58.00	90.00	7.35	0.52	0.18	105.00	158.00	0.39	0.27	58.00
3.00	55.00	53.00	57.23	0.32	0.16	215.00	88.00	0.25	0.15	47.00
4.00	44.00	109.00	43.37	0.55	0.07	136.00	108.00	0.17	0.05	60.00
5.00	56.00	64.00	11.97	0.12	0.08	88.00	105.00	0.32	0.23	45.00
6.00	39.00	112.00	14.74	0.58	0.10	178.00	85.00	0.44	0.34	33.00
7.00	49.00	35.00	32.29	0.13	0.14	119.00	220.00	0.24	0.06	43.00
8.00	52.00	95.00	55.38	0.41	0.19	226.00	187.00	0.06	0.10	39.00
9.00	46.00	18.00	1.80	0.91	0.20	170.00	130.00	0.09	0.30	39.00
10.00	59.00	87.00	29.52	0.87	0.04	111.00	226.00	0.22	0.26	30.00
11.00	38.00	17.00	58.15	0.63	0.12	139.00	116.00	0.36	0.01	37.00
12.00	59.00	69.00	41.53	1.02	0.11	71.00	128.00	0.42	0.12	41.00
13.00	40.00	38.00	20.28	0.74	0.20	147.00	94.00	0.18	0.36	59.00
14.00	50.00	72.00	25.82	0.99	0.15	234.00	68.00	0.05	0.20	46.00
15.00	41.00	49.00	47.07	1.05	0.22	99.00	144.00	0.47	0.05	55.00
16.00	45.00	76.00	33.21	0.82	0.10	232.00	240.00	0.30	0.17	54.00
17.00	54.00	61.00	23.97	0.23	0.31	240.00	201.00	0.49	0.22	48.00
18.00	40.00	105.00	16.58	0.44	0.27	142.00	209.00	0.41	0.14	52.00
19.00	48.00	56.00	34.14	0.40	0.35	220.00	66.00	0.23	0.20	54.00
20.00	42.00	92.00	51.69	0.16	0.26	91.00	139.00	0.13	0.31	56.00
21.00	52.00	36.00	24.90	0.19	0.43	128.00	63.00	0.37	0.11	33.00
22.00	51.00	102.00	0.88	0.26	0.25	187.00	147.00	0.45	0.07	40.00
23.00	60.00	58.00	47.99	0.35	0.43	83.00	189.00	0.11	0.21	40.00
24.00	42.00	120.00	49.84	0.48	0.32	156.00	164.00	0.15	0.38	32.00
25.00	43.00	41.00	6.42	0.69	0.37	192.00	229.00	0.12	0.04	35.00
26.00	49.00	81.00	10.12	1.10	0.35	94.00	175.00	0.21	0.14	44.00
27.00	53.00	25.00	45.22	0.90	0.25	223.00	119.00	0.48	0.29	38.00
28.00	56.00	104.00	37.83	0.93	0.41	116.00	83.00	0.35	0.28	41.00
29.00	57.00	46.00	8.27	0.71	0.30	97.00	133.00	0.09	0.08	56.00
30.00	47.00	113.00	21.20	0.83	0.40	237.00	122.00	0.29	0.02	53.00
31.00	45.00	28.00	42.45	0.66	0.42	133.00	198.00	0.39	0.35	59.00
32.00	54.00	84.00	56.31	0.96	0.33	175.00	223.00	0.35	0.25	48.00
33.00	38.00	68.00	30.44	0.60	0.24	150.00	150.00	0.28	0.19	45.00
34.00	28.00	115.00	38.75	0.77	0.39	102.00	97.00	0.28	0.01	38.00
35.00	17.00	45.00	53.53	0.68	0.30	195.00	142.00	0.16	0.11	32.00
36.00	20.00	82.00	3.65	0.88	0.32	85.00	212.00	0.30	0.23	43.00
37.00	31.00	26.00	17.51	0.65	0.42	164.00	192.00	0.38	0.33	30.00
38.00	19.00	71.00	48.92	1.08	0.40	212.00	195.00	0.23	0.15	45.00
39.00	36.00	23.00	46.14	0.62	0.38	122.00	215.00	0.11	0.04	57.00
40.00	26.00	100.00	28.59	1.07	0.34	181.00	80.00	0.31	0.32	47.00
41.00	23.00	40.00	5.50	0.79	0.29	74.00	113.00	0.49	0.29	51.00
42.00	29.00	117.00	59.08	0.29	0.28	130.00	170.00	0.46	0.08	51.00
43.00	16.00	48.00	31.36	0.33	0.44	189.00	74.00	0.33	0.12	60.00
44.00	37.00	118.00	2.73	0.57	0.36	161.00	184.00	0.19	0.37	53.00
45.00	16.00	66.00	19.36	0.18	0.37	229.00	173.00	0.13	0.26	49.00
46.00	35.00	97.00	40.60	0.46	0.28	153.00	206.00	0.37	0.02	31.00
47.00	25.00	63.00	35.06	0.21	0.33	66.00	232.00	0.50	0.18	44.00
48.00	34.00	86.00	13.81	0.15	0.27	201.00	156.00	0.08	0.33	35.00
49.00	30.00	59.00	27.67	0.38	0.38	68.00	60.00	0.25	0.21	36.00
50.00	21.00	74.00	36.91	0.98	0.17	60.00	99.00	0.06	0.16	42.00
51.00	35.00	30.00	44.30	0.76	0.21	158.00	91.00	0.14	0.24	38.00
52.00	27.00	79.00	26.75	0.80	0.13	80.00	234.00	0.32	0.18	36.00
53.00	33.00	43.00	9.19	1.04	0.22	209.00	161.00	0.42	0.07	34.00
54.00	23.00	99.00	35.98	1.01	0.05	173.00	237.00	0.18	0.27	57.00
55.00	24.00	33.00	60.00	0.94	0.23	113.00	153.00	0.10	0.31	50.00
56.00	15.00	77.00	12.89	0.85	0.05	218.00	111.00	0.44	0.17	50.00
57.00	33.00	15.00	11.04	0.73	0.17	144.00	136.00	0.40	0.00	58.00
58.00	32.00	94.00	54.46	0.51	0.12	108.00	71.00	0.43	0.34	55.00
59.00	26.00	54.00	50.76	0.10	0.13	206.00	125.00	0.34	0.24	46.00
60.00	22.00	110.00	15.66	0.30	0.23	77.00	181.00	0.07	0.09	53.00
61.00	19.00	31.00	23.05	0.27	0.07	184.00	218.00	0.20	0.10	49.00
62.00	18.00	89.00	52.61	0.49	0.18	203.00	167.00	0.46	0.30	34.00
63.00	28.00	22.00	39.68	0.37	0.08	63.00	178.00	0.26	0.36	37.00
64.00	30.00	107.00	18.43	0.54	0.06	167.00	102.00	0.16	0.03	31.00
65.00	21.00	51.00	4.58	0.24	0.15	125.00	77.00	0.20	0.13	42.00

3. Littoral Transit Simulation Run

Param. Grp.	fix interval	SU_IATime	TimeBtwnInterrupt	IntLengthMean	InterLengthSigma	TimeBtwnCrseChg	TimeBtwnSpdChg	PrCrseChg	PrSpdChg	RadioInt
1.00	22.00	3.00	22.13	0.43	0.09	32.00	49.00	0.46	0.87	23.00
2.00	29.00	11.00	7.35	0.52	0.18	14.00	35.00	0.71	0.66	28.00
3.00	27.00	7.00	57.23	0.32	0.16	35.00	14.00	0.43	0.40	19.00
4.00	20.00	14.00	43.37	0.55	0.07	20.00	20.00	0.25	0.16	30.00
5.00	28.00	8.00	11.97	0.12	0.08	10.00	19.00	0.58	0.57	17.00
6.00	17.00	14.00	14.74	0.58	0.10	28.00	13.00	0.82	0.81	8.00
7.00	24.00	4.00	32.29	0.13	0.14	16.00	54.00	0.40	0.18	16.00
8.00	25.00	12.00	55.38	0.41	0.19	37.00	44.00	0.01	0.26	13.00
9.00	22.00	2.00	1.80	0.91	0.20	26.00	26.00	0.09	0.73	12.00
10.00	29.00	11.00	29.52	0.87	0.04	15.00	56.00	0.36	0.62	5.00
11.00	17.00	2.00	58.15	0.63	0.12	20.00	22.00	0.65	0.06	11.00
12.00	30.00	9.00	41.53	1.02	0.11	7.00	26.00	0.77	0.32	14.00
13.00	18.00	5.00	20.28	0.74	0.20	22.00	15.00	0.27	0.86	29.00
14.00	24.00	9.00	25.82	0.99	0.15	39.00	8.00	0.00	0.49	18.00
15.00	19.00	6.00	47.07	1.05	0.22	13.00	31.00	0.89	0.17	26.00
16.00	21.00	10.00	33.21	0.82	0.10	38.00	60.00	0.53	0.44	25.00
17.00	26.00	8.00	23.97	0.23	0.31	40.00	48.00	0.92	0.54	20.00
18.00	18.00	13.00	16.58	0.44	0.27	21.00	51.00	0.76	0.36	23.00
19.00	23.00	7.00	34.14	0.40	0.35	36.00	7.00	0.39	0.50	25.00
20.00	19.00	12.00	51.69	0.16	0.26	11.00	29.00	0.16	0.74	27.00
21.00	25.00	5.00	24.90	0.19	0.43	18.00	6.00	0.67	0.30	7.00
22.00	25.00	13.00	0.88	0.26	0.25	30.00	32.00	0.85	0.20	14.00
23.00	30.00	7.00	47.99	0.35	0.43	9.00	45.00	0.12	0.53	13.00
24.00	19.00	15.00	49.84	0.48	0.32	24.00	37.00	0.21	0.90	7.00
25.00	20.00	5.00	6.42	0.69	0.37	31.00	57.00	0.15	0.13	9.00
26.00	23.00	10.00	10.12	1.10	0.35	12.00	40.00	0.34	0.37	16.00
27.00	26.00	3.00	45.22	0.90	0.25	37.00	23.00	0.91	0.70	11.00
28.00	27.00	13.00	37.83	0.93	0.41	16.00	12.00	0.62	0.67	14.00
29.00	28.00	6.00	8.27	0.71	0.30	12.00	27.00	0.07	0.24	26.00
30.00	22.00	14.00	21.20	0.83	0.40	39.00	24.00	0.50	0.10	25.00
31.00	21.00	4.00	42.45	0.66	0.42	19.00	47.00	0.73	0.83	29.00
32.00	27.00	11.00	56.31	0.96	0.33	27.00	55.00	0.64	0.61	20.00
33.00	17.00	9.00	30.44	0.60	0.24	23.00	33.00	0.48	0.48	18.00
34.00	11.00	14.00	38.75	0.77	0.39	13.00	16.00	0.49	0.08	12.00
35.00	4.00	6.00	53.53	0.68	0.30	31.00	30.00	0.24	0.29	7.00
36.00	6.00	10.00	3.65	0.88	0.32	10.00	51.00	0.52	0.55	16.00
37.00	13.00	3.00	17.51	0.65	0.42	25.00	45.00	0.70	0.79	5.00
38.00	5.00	9.00	48.92	1.08	0.40	35.00	46.00	0.37	0.38	18.00
39.00	16.00	3.00	46.14	0.62	0.38	17.00	52.00	0.13	0.14	27.00
40.00	9.00	13.00	28.59	1.07	0.34	29.00	11.00	0.55	0.77	19.00
41.00	8.00	5.00	5.50	0.79	0.29	8.00	21.00	0.94	0.69	22.00
42.00	11.00	15.00	59.08	0.29	0.28	19.00	39.00	0.86	0.22	23.00
43.00	4.00	6.00	31.36	0.33	0.44	30.00	9.00	0.59	0.33	30.00
44.00	16.00	15.00	2.73	0.57	0.36	25.00	43.00	0.30	0.89	24.00
45.00	3.00	8.00	19.36	0.18	0.37	38.00	39.00	0.18	0.63	21.00
46.00	15.00	12.00	40.60	0.46	0.28	23.00	50.00	0.68	0.09	6.00
47.00	9.00	8.00	35.06	0.21	0.33	6.00	57.00	0.95	0.46	17.00
48.00	14.00	11.00	13.81	0.15	0.27	32.00	34.00	0.06	0.78	9.00
49.00	12.00	7.00	27.67	0.38	0.38	7.00	5.00	0.42	0.51	10.00
50.00	7.00	9.00	36.91	0.98	0.17	5.00	17.00	0.03	0.41	15.00
51.00	15.00	4.00	44.30	0.76	0.21	24.00	14.00	0.19	0.59	12.00
52.00	10.00	10.00	26.75	0.80	0.13	9.00	58.00	0.56	0.45	10.00
53.00	14.00	5.00	9.19	1.04	0.22	34.00	36.00	0.79	0.21	8.00
54.00	8.00	12.00	35.98	1.01	0.05	27.00	59.00	0.28	0.65	28.00
55.00	8.00	4.00	60.00	0.94	0.23	15.00	33.00	0.10	0.75	21.00
56.00	3.00	10.00	12.89	0.85	0.05	36.00	20.00	0.83	0.42	22.00
57.00	14.00	2.00	11.04	0.73	0.17	21.00	28.00	0.74	0.05	28.00
58.00	13.00	12.00	54.46	0.51	0.12	14.00	8.00	0.80	0.82	26.00
59.00	10.00	7.00	50.76	0.10	0.13	33.00	25.00	0.61	0.58	19.00
60.00	7.00	14.00	15.66	0.30	0.23	8.00	42.00	0.04	0.25	24.00
61.00	6.00	4.00	23.05	0.27	0.07	29.00	53.00	0.33	0.28	21.00
62.00	5.00	11.00	52.61	0.49	0.18	33.00	38.00	0.88	0.71	9.00
63.00	11.00	3.00	39.68	0.37	0.08	6.00	41.00	0.45	0.85	10.00
64.00	12.00	13.00	18.43	0.54	0.06	26.00	18.00	0.22	0.12	6.00
65.00	6.00	6.00	4.58	0.24	0.15	18.00	10.00	0.31	0.34	15.00

4. Anti Surface Warfare Simulation Run

Param. Grp.	fix interval	SU_IATime	TimeBtwnInterrupt	IntLengthMean	InterLengthSigma	TimeBtwnCrseChg	TimeBtwnSpdChg	PrCrseChg	PrSpdChg	RadioInt	PrEngage
1	22	3.00	22.13	0.43	0.09	13.00	13.00	0.27	0.37	23.00	0.59
2	29	11.00	7.35	0.52	0.18	8.00	10.00	0.39	0.27	28.00	0.79
3	27	7.00	57.23	0.32	0.16	14.00	7.00	0.25	0.15	19.00	0.82
4	20	14.00	43.37	0.55	0.07	9.00	8.00	0.17	0.05	30.00	0.92
5	28	8.00	11.97	0.12	0.08	7.00	8.00	0.32	0.23	17.00	0.16
6	17	14.00	14.74	0.58	0.10	12.00	6.00	0.44	0.34	8.00	0.47
7	24	4.00	32.29	0.13	0.14	8.00	14.00	0.24	0.06	16.00	0.41
8	25	12.00	55.38	0.41	0.19	14.00	12.00	0.06	0.10	13.00	0.11
9	22	2.00	1.80	0.91	0.20	11.00	9.00	0.09	0.30	12.00	0.86
10	29	11.00	29.52	0.87	0.04	8.00	14.00	0.22	0.26	5.00	0.70
11	17	2.00	58.15	0.63	0.12	9.00	8.00	0.36	0.01	11.00	0.75
12	30	9.00	41.53	1.02	0.11	6.00	9.00	0.42	0.12	14.00	0.58
13	18	5.00	20.28	0.74	0.20	10.00	7.00	0.18	0.36	29.00	0.23
14	24	9.00	25.82	0.99	0.15	15.00	5.00	0.05	0.20	18.00	0.33
15	19	6.00	47.07	1.05	0.22	7.00	10.00	0.47	0.05	26.00	0.10
16	21	10.00	33.21	0.82	0.10	15.00	15.00	0.30	0.17	25.00	0.42
17	26	8.00	23.97	0.23	0.31	15.00	13.00	0.49	0.22	20.00	0.38
18	18	13.00	16.58	0.44	0.27	10.00	13.00	0.41	0.14	23.00	0.20
19	23	7.00	34.14	0.40	0.35	14.00	5.00	0.23	0.20	25.00	0.54
20	19	12.00	51.69	0.16	0.26	7.00	9.00	0.13	0.31	27.00	0.14
21	25	5.00	24.90	0.19	0.43	9.00	5.00	0.37	0.11	7.00	0.48
22	25	13.00	0.88	0.26	0.25	12.00	10.00	0.45	0.07	14.00	0.93
23	30	7.00	47.99	0.35	0.43	6.00	12.00	0.11	0.21	13.00	0.61
24	19	15.00	49.84	0.48	0.32	10.00	11.00	0.15	0.38	7.00	0.73
25	20	5.00	6.42	0.69	0.37	12.00	14.00	0.12	0.04	9.00	0.34
26	23	10.00	10.12	1.10	0.35	7.00	11.00	0.21	0.14	16.00	0.13
27	26	3.00	45.22	0.90	0.25	14.00	8.00	0.48	0.29	11.00	0.30
28	27	13.00	37.83	0.93	0.41	8.00	6.00	0.35	0.28	14.00	0.44
29	28	6.00	8.27	0.71	0.30	7.00	9.00	0.09	0.08	26.00	0.89
30	22	14.00	21.20	0.83	0.40	15.00	8.00	0.29	0.02	25.00	0.65
31	21	4.00	42.45	0.66	0.42	9.00	13.00	0.39	0.35	29.00	0.83
32	27	11.00	56.31	0.96	0.33	11.00	14.00	0.35	0.25	20.00	0.85
33	17	9.00	30.44	0.60	0.24	10.00	10.00	0.28	0.19	18.00	0.55
34	11	14.00	38.75	0.77	0.39	7.00	7.00	0.28	0.01	12.00	0.51
35	4	6.00	53.53	0.68	0.30	13.00	10.00	0.16	0.11	7.00	0.31
36	6	10.00	3.65	0.88	0.32	6.00	13.00	0.30	0.23	16.00	0.28
37	13	3.00	17.51	0.65	0.42	11.00	12.00	0.38	0.33	5.00	0.18
38	5	9.00	48.92	1.08	0.40	13.00	13.00	0.23	0.15	18.00	0.94
39	16	3.00	46.14	0.62	0.38	8.00	14.00	0.11	0.04	27.00	0.63
40	9	13.00	28.59	1.07	0.34	12.00	6.00	0.31	0.32	19.00	0.69
41	8	5.00	5.50	0.79	0.29	6.00	8.00	0.49	0.29	22.00	0.99
42	11	15.00	59.08	0.29	0.28	9.00	11.00	0.46	0.08	23.00	0.24
43	4	6.00	31.36	0.33	0.44	12.00	6.00	0.33	0.12	30.00	0.40
44	16	15.00	2.73	0.57	0.36	11.00	12.00	0.19	0.37	24.00	0.35
45	3	8.00	19.36	0.18	0.37	14.00	11.00	0.13	0.26	21.00	0.52
46	15	12.00	40.60	0.46	0.28	10.00	13.00	0.37	0.02	6.00	0.87
47	9	8.00	35.06	0.21	0.33	5.00	15.00	0.50	0.18	17.00	0.78
48	14	11.00	13.81	0.15	0.27	13.00	10.00	0.08	0.33	9.00	1.00
49	12	7.00	27.67	0.38	0.38	5.00	5.00	0.25	0.21	10.00	0.68
50	7	9.00	36.91	0.98	0.17	5.00	7.00	0.06	0.16	15.00	0.72
51	15	4.00	44.30	0.76	0.21	10.00	7.00	0.14	0.24	12.00	0.90
52	10	10.00	26.75	0.80	0.13	6.00	15.00	0.32	0.18	10.00	0.56
53	14	5.00	9.19	1.04	0.22	13.00	11.00	0.42	0.07	8.00	0.96
54	8	12.00	35.98	1.01	0.05	11.00	15.00	0.18	0.27	28.00	0.62
55	8	4.00	60.00	0.94	0.23	8.00	10.00	0.10	0.31	21.00	0.17
56	3	10.00	12.89	0.85	0.05	14.00	8.00	0.44	0.17	22.00	0.49
57	14	2.00	11.04	0.73	0.17	10.00	9.00	0.40	0.00	28.00	0.37
58	13	12.00	54.46	0.51	0.12	8.00	6.00	0.43	0.34	26.00	0.76
59	10	7.00	50.76	0.10	0.13	13.00	9.00	0.34	0.24	19.00	0.97
60	7	14.00	15.66	0.30	0.23	6.00	12.00	0.07	0.09	24.00	0.80
61	6	4.00	23.05	0.27	0.07	12.00	14.00	0.20	0.10	21.00	0.66
62	5	11.00	52.61	0.49	0.18	13.00	11.00	0.46	0.30	9.00	0.21
63	11	3.00	39.68	0.37	0.08	5.00	12.00	0.26	0.36	10.00	0.45
64	12	13.00	18.43	0.54	0.06	11.00	7.00	0.16	0.03	6.00	0.27
65	6	6.00	4.58	0.24	0.15	9.00	6.00	0.20	0.13	15.00	0.25

5. Naval Surface Fire Support Simulation Run

Param. Grp.	fix interval	SU_IATime	TimeBtwnInterrupt	IntLengthMean	InterLengthSigma	TimeBtwnCrseChg	TimeBtwnSpdChg	TimeBtwnStrike	PrCrseChg	PrSpdChg	RadioInt
1	12.00	7.00	22.13	0.43	0.09	32.00	49.00	16.00	0.49	0.27	19.00
2	14.00	34.00	7.35	0.52	0.18	14.00	35.00	23.00	0.37	0.35	24.00
3	14.00	19.00	57.23	0.32	0.16	35.00	14.00	15.00	0.23	0.22	25.00
4	11.00	41.00	43.37	0.55	0.07	20.00	20.00	9.00	0.11	0.37	28.00
5	14.00	24.00	11.97	0.12	0.08	10.00	19.00	19.00	0.32	0.18	7.00
6	9.00	42.00	14.74	0.58	0.10	28.00	13.00	26.00	0.45	0.04	15.00
7	12.00	13.00	32.29	0.13	0.14	16.00	54.00	14.00	0.12	0.17	14.00
8	13.00	36.00	55.38	0.41	0.19	37.00	44.00	2.00	0.16	0.12	5.00
9	11.00	6.00	1.80	0.91	0.20	26.00	26.00	5.00	0.41	0.11	26.00
10	15.00	33.00	29.52	0.87	0.04	15.00	56.00	13.00	0.35	0.00	22.00
11	9.00	6.00	58.15	0.63	0.12	20.00	22.00	21.00	0.06	0.09	23.00
12	15.00	26.00	41.53	1.02	0.11	7.00	26.00	25.00	0.19	0.14	18.00
13	10.00	14.00	20.28	0.74	0.20	22.00	15.00	10.00	0.48	0.36	9.00
14	12.00	27.00	25.82	0.99	0.15	39.00	8.00	2.00	0.28	0.20	11.00
15	10.00	18.00	47.07	1.05	0.22	13.00	31.00	28.00	0.11	0.31	5.00
16	11.00	28.00	33.21	0.82	0.10	38.00	60.00	18.00	0.25	0.31	14.00
17	13.00	23.00	23.97	0.23	0.31	40.00	48.00	29.00	0.31	0.23	13.00
18	10.00	39.00	16.58	0.44	0.27	21.00	51.00	24.00	0.21	0.28	8.00
19	12.00	21.00	34.14	0.40	0.35	36.00	7.00	13.00	0.29	0.30	17.00
20	10.00	34.00	51.69	0.16	0.26	11.00	29.00	7.00	0.42	0.33	6.00
21	13.00	13.00	24.90	0.19	0.43	18.00	6.00	22.00	0.18	0.04	16.00
22	13.00	38.00	0.88	0.26	0.25	30.00	32.00	27.00	0.13	0.13	28.00
23	15.00	21.00	47.99	0.35	0.43	9.00	45.00	6.00	0.30	0.12	19.00
24	10.00	45.00	49.84	0.48	0.32	24.00	37.00	8.00	0.50	0.02	23.00
25	11.00	15.00	6.42	0.69	0.37	31.00	57.00	6.00	0.09	0.06	12.00
26	12.00	30.00	10.12	1.10	0.35	12.00	40.00	12.00	0.22	0.17	6.00
27	13.00	9.00	45.22	0.90	0.25	37.00	23.00	29.00	0.39	0.10	10.00
28	14.00	39.00	37.83	0.93	0.41	16.00	12.00	20.00	0.38	0.14	14.00
29	14.00	17.00	8.27	0.71	0.30	12.00	27.00	4.00	0.15	0.33	27.00
30	11.00	43.00	21.20	0.83	0.40	39.00	24.00	17.00	0.08	0.30	20.00
31	11.00	10.00	42.45	0.66	0.42	19.00	47.00	23.00	0.46	0.37	25.00
32	14.00	31.00	56.31	0.96	0.33	27.00	55.00	21.00	0.35	0.23	26.00
33	9.00	25.00	30.44	0.60	0.24	23.00	33.00	16.00	0.28	0.19	18.00
34	6.00	43.00	38.75	0.77	0.39	13.00	16.00	16.00	0.06	0.11	16.00
35	4.00	16.00	53.53	0.68	0.30	31.00	30.00	9.00	0.18	0.03	11.00
36	4.00	31.00	3.65	0.88	0.32	10.00	51.00	17.00	0.32	0.16	10.00
37	7.00	9.00	17.51	0.65	0.42	25.00	45.00	23.00	0.44	0.01	7.00
38	4.00	26.00	48.92	1.08	0.40	35.00	46.00	13.00	0.23	0.20	28.00
39	9.00	8.00	46.14	0.62	0.38	17.00	52.00	6.00	0.10	0.34	20.00
40	6.00	38.00	28.59	1.07	0.34	29.00	11.00	18.00	0.43	0.21	21.00
41	5.00	14.00	5.50	0.79	0.29	8.00	21.00	30.00	0.39	0.26	30.00
42	7.00	44.00	59.08	0.29	0.28	19.00	39.00	27.00	0.14	0.27	9.00
43	3.00	18.00	31.36	0.33	0.44	30.00	9.00	20.00	0.20	0.38	13.00
44	9.00	44.00	2.73	0.57	0.36	25.00	43.00	11.00	0.49	0.29	12.00
45	3.00	24.00	19.36	0.18	0.37	38.00	39.00	7.00	0.36	0.24	17.00
46	8.00	36.00	40.60	0.46	0.28	23.00	50.00	22.00	0.07	0.02	26.00
47	6.00	23.00	35.06	0.21	0.33	6.00	57.00	30.00	0.27	0.18	24.00
48	8.00	32.00	13.81	0.15	0.27	32.00	34.00	4.00	0.44	0.07	30.00
49	7.00	22.00	27.67	0.38	0.38	7.00	5.00	14.00	0.30	0.07	21.00
50	5.00	28.00	36.91	0.98	0.17	5.00	17.00	3.00	0.24	0.15	22.00
51	8.00	11.00	44.30	0.76	0.21	24.00	14.00	8.00	0.34	0.10	27.00
52	6.00	29.00	26.75	0.80	0.13	9.00	58.00	19.00	0.26	0.08	18.00
53	8.00	16.00	9.19	1.04	0.22	34.00	36.00	25.00	0.13	0.05	29.00
54	5.00	37.00	35.98	1.01	0.05	27.00	59.00	10.00	0.37	0.34	19.00
55	5.00	12.00	60.00	0.94	0.23	15.00	33.00	5.00	0.42	0.25	7.00
56	3.00	29.00	12.89	0.85	0.05	36.00	20.00	27.00	0.25	0.26	16.00
57	8.00	5.00	11.04	0.73	0.17	21.00	28.00	24.00	0.05	0.36	12.00
58	8.00	35.00	54.46	0.51	0.12	14.00	8.00	26.00	0.46	0.32	23.00
59	6.00	20.00	50.76	0.10	0.13	33.00	25.00	20.00	0.33	0.21	29.00
60	5.00	41.00	15.66	0.30	0.23	8.00	42.00	3.00	0.16	0.29	25.00
61	4.00	11.00	23.05	0.27	0.07	29.00	53.00	12.00	0.17	0.24	21.00
62	4.00	33.00	52.61	0.49	0.18	33.00	38.00	28.00	0.40	0.05	8.00
63	7.00	8.00	39.68	0.37	0.08	6.00	41.00	15.00	0.47	0.08	15.00
64	7.00	40.00	18.43	0.54	0.06	26.00	18.00	9.00	0.09	0.01	10.00
65	5.00	19.00	4.58	0.24	0.15	18.00	10.00	11.00	0.20	0.15	9.00

LIST OF REFERENCES

Baitis, A.E., Applebee, T.R., McNamara, T.M., (1984). Human Factors Considerations Applied to Operations of the FFG-8 and LAMPS MK III. *Naval Engineers Journal*. 97(4)

Baitis A.E., Holcombe F.D., Conwell S.L., Crossland P., Colwell J., Pattison J.H., Strong R., (1995). Motion Induced Interruption (MII) and Motion Induced Fatigue (MIF) Experiments at the Naval Biodynamics Laboratory. CRDKNSWC-HD-1423-01. Maryland, Carderock Division, Naval Surface Warfare Center.

Benson, A.J., (1984). Motion Sickness. *Vertigo*. New York, Wiley.

Buss, A., (2001). Basic Event Graph Modeling. *Simulation News Europe: Technical Notes*. 31, 1-6.

Buss, A., (2002). Component Based Simulation Modeling With Simkit. *Proceedings of the 2002 Winter Simulation Conference.* California, Naval Postgraduate School.

Buss, A., Sanchez, P., (2002). Building Complex Models With LEGOS (Listener Event Graph Objects). *Proceedings of the 2002 Winter Simulation Conference.* California, Naval Postgraduate School.

Colwell, J.L., (1989). Human Factors in a Naval Environment: A Review of motion Sickness and Biodynamic Problems. Technical Memorandum 89/220. Canada, Defense Research Establishment, Atlantic.

Crossland, P., (1998). A Rational Approach to Specifying Seakeeping Performance in the Ship Design Process. Warship, RINA International Symposium.

Department of the Navy (2006). Memorandum for Joint Speed Vessel (JHSV) Analysis of Alternatives (AoA) Study. (2005, May 06).

Donohew, B.E., Griffin, M.J., (2004). Motion sickness: effect of the frequency of lateral oscillation. *Aviation, Space, and Environmental Medicine.* 75, 649 –56.

Gianaros, P., Muth, E., Mordkoff, J.T., Levine, M., Stern, R.M. (2001). A Questioner for the Assessment of the Multiple Dimension of Motion Sickness. *Aviation, Space, and Environmental Medicine.* 72(2), 115-119.

Gourley, S., Scott, R. (2005). Speed At Sea, the Key for Intra-Theatre Lift. *Jane's Navy International Online Magazine*. Retrieved June 2005 from <http://jni.janes.com/>.

Graham, R., (1990). Motion-Induced Interruptions as Ship Operability Criteria. *American Society of Naval Engineers Journal*. 102(2).

Graham R., Baitis A.E., Meyers, W.G., (1991). A Frequency Domain Method for Estimating the Incidence and Severity of Sliding. DTRC/SHD-1361-01. Maryland, David Taylor Research Center.

Graybiel, A., Wood, C.D., Mille,r E.F., Cramer, D.B., (1968). Diagnostic criteria for Grading the severity of acute motion sickness. *Aerospace Medicine.* 39, 453-455.

Griffin, M.J., (1990). *Handbook of Human Vibration*. New York, Academic Press.

Kellogg, R.S., Kennedy, R.S., Graybiel, A., (1965). Motion sickness Symptomatology of Labyrinthine Defective and Normal Subjects During Zero Gravity Maneuvers. *Aerospace Medicine.* 36, 315-318.

Lawther, A., Griffin, M.J., (1998). Prediction of the Incidence of Motion Sickness from the Magnitude, Frequency, and Duration of Vertical Oscillations. *Journal of the Acoustical Society of America.* 82(3), 957-966.

LCS Will Transform Naval Operations in the Littorals. (2005). PEO Ships Official U.S. Navy Website. Retrieved November 11, 2005 from <http://peoships.crane.navy.mil/lcs/default.htm>.

Littoral Combat Ship (LCS) High Speed Surface Ship, USA. (2005). Naval Technology Website. Retrieved November 11, 2005 from <http://www.naval-technology.com/projects/littoral/>.

Mansfield, N.J., (2005). *Human Response to Vibration.* New Jersey, CRC Press.

McCauley, M.E., Jackson, W.R., Wylie, C.D., O'Hanlon, J.F., Mackie, R.R., (1976). Motion Sickness Incidence: Exploratory Studies of Habituation, Pitch, and Roll, and the Refinement of a Mathematical Model. Technical Report No. 1733-2. Virginia, Office of Naval Research.

Money, K.E., (1970). Motion Sickness. *Physiological Reviews*. 50(1), 1-39.

Montgomery, D.C., Peck, E.A., Vining, G.G., (2001). *Introduction to Linear Regression Analysis*. 3rd Ed. New York. John Wiley & Sons Inc.

Reason, J.T., Brand, J.J., (1975). *Motion Sickness*. London, Academic Press.

Rose, R.G., (2005). HSV Operation in the Western Pacific. *Marine Corps Gazette*. 89(3), 60-63.

Sanchez, S., (2005). Work Smarter, Not Harder: Guidelines for Designing Simulation Experiments. *Proceedings of the 2005 Winter Simulation Conference.* California, Naval Postgraduate School.

Smart, L.J., Stoffregen, T.A., Bardy, B.G., (2002). Visually Induced Motion Sickness Predicted by Postural Instability. *Human Factors.* 44(3)

Stevens, S.C., Parson M.G., (2002). Effects of Motion at Sea On Crew Performance: A Survey. *Marine Technology.* 39(1), 29-47.

United States Marine Corps (USMC) High Speed Connector Project. (2005). USMC HSC Website. Retrieved November 28, 2005 from <http://www.hscproject.net>.

Wickens, C., Lee, J., Liu, Y., Becker, G., (2004). *An Introduction to Human Factors Engineering.* Second Edition. New Jersey, Pearson Education, Inc.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Robert McDonald
   Naval Surface Warfare Center Code A82
   Panama City, Florida

4. Michael McCauley
   Naval Postgraduate School
   Monterey, California

5. Lyn Whitaker
   Naval Postgraduate School
   Monterey, California

6. Arnold Buss
   Naval Postgraduate School
   Monterey, California